Commodore
COMPUTER CLUB

67 L. 5.000

La rivista degli utenti di sistemi Commodore

Anno VIII - N. 67 - 25 Settembre 1989 - Sped. Abb. Post. Gr. III/70 - CR - Distr. MePe

C 128: un turbo per file sequenziali

Protezioni: un programma anti-cartuccia

Assembler Amiga

Coloriamo il pointer (e non solo...)

Assembler '64

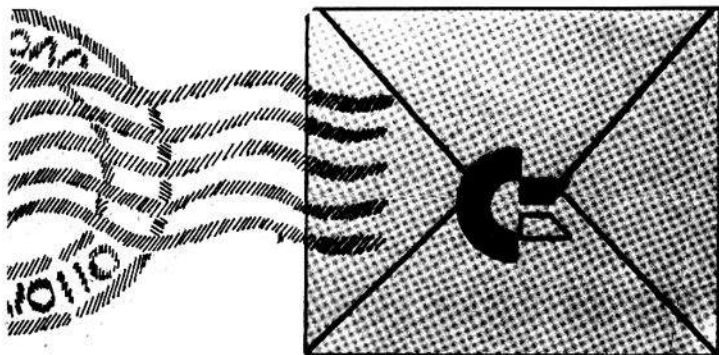
Il '64 emula il multitasking dell'Amiga

Sommario

RUBRICHE

4	LA VOSTRA POSTA
67	ANNUNCI
77	VIDEOGAMES
81	GUIDA ALL'ACQUISTO
84	I COMMODORE POINTS
86	SYSTEMS EDITORIALE PER TE

PAG.	REMarks	C64	C128	C16	Amiga	Gener.
	Hardware					
10	Da 500 a 2000 il passo è breve				•	
	Protezioni					
13	Firmate i vostri dischetti				•	
61	Davide contro Golia	•				
	Telematica					
18	Pronto, chi byte					•
	Spazio 128					
23	Sequenziali a tutto gas		•			
69	Dieci, cento, mille editor		•			
CAMPUS: inserto speciale per piccoli Commodore						
29/I	I cicli del 6502	•				
34/VI	Un solo cervello per mille funzioni	•				
45/XVII	Tutto il macroassembler che volete	•				
50/XXII	Un pointer in techniclor				•	
56/XXVIII	Un riempimento personalizzato	•				

Direttore: Alessandro de Simone - **Caporedattore:** Michele Maggi
Redazione/collaboratori: Paolo Agostini, Davide Ardizzone, Claudio Baiocchi, Angelo Bianchi, Luigi Callegari, Sergio Camici, Umberto Colapicchioni, Laura & Miria Colombo, Maurizio Dell'Abate, Valerio Ferri, Simona Locati, Cristina Magnaghi, Giancarlo Mariani, Roberto Marigo, Clizio Merli, Marco Mietta, Marco Miotti, Oscar Moccia, Roberto Morassi, Guido Pagani, Antonio Pastorelli, Domenico Pavone, Armando Sforzi, Sonja Scharrer, Fabio Sorgato, Valentino Spataro, Danilo Toma, Franco Rodella, Stefano Simonelli
Grafica: Arturo Ciaglia
Direzione, pubblicità: via Mosè, 22 - 20090 Opera (MI) - Tel. 02/55500310 - **Redazione:** Tel. 02/5249211
Pubblicità: Milano: Leandro Nencioni (direttore vendite), - Via Mosè, 22 - 20090 Opera (MI) - Tel. 02/55500310
• Emilia Romagna: Spazio E - P.zza Roosevelt, 4 - 40123 Bologna - Tel. 051/236979
• Toscana, Marche, Umbria: Mercurio srl - via Rodari, 9 - San Giovanni Valdarno (Ar) - Tel. 055/947444
• Lazio, Campania: Spazio Nuovo - via P. Foscari, 70 - 00139 Roma - Tel. 06/8109679
Segreteria: Tiziana Sodano - **Abbonamenti:** Liliana Spina
Arretrati e software: Milano, V.le Famagosta, 75 - tel. 02/8467348 - Sig.ra Lucia Dominoni (il servizio è operativo nelle ore pomeridiane. Nelle altre ore rivolgersi allo 02/55500310)
Tariffe: prezzo per copia L. 5.000. Abbonamento annuo (11 fascicoli) L. 50.000. Estero: il doppio.
Abbonamento cumulativo alle riviste Computer e Commodore Computer Club L. 90.000.
I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 37952207
Composizione: Systems Editoriale Srl - **Fotolito:** Systems Editoriale Srl
Stampa: Systems Editoriale/La Litografica Srl - Busto Arsizio (Va)
Registrazioni: Tribunale di Milano n. 370 del 2/10/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70% - **Distrib.:** MePe - via G. Carcano, 32 - Milano
Periodici Systems: Banca Oggi - Commodore Club (disco) - Commodore Computer Club - Commodore Computer Club (disco produzione tedesca) - Computer - Computer disco - Electronic Mass Media Age - Energy Manager - Hospital Management - Jonathan - MondoRicambi - Nursing '90 - PC Programm (disco) - Personal Computer - Security - Software Club (cassetta ed. italiana) - TuttoGatto Videoteca - VR Videoregistrare

la vostra posta

CAMBIO PAGINA

□ Uso *Easy Script* con stampante Mps-1200 ed utilizzo a volte il modulo di carta continuo, a volte i fogli singoli. Solo in quest'ultimo caso, dopo aver sostituito il foglio di carta, il computer mi segnala un errore e blocca la procedura di stampa. Quale può essere il difetto?
(Vito Catania - Catania)

• Una *qualsiasi* stampante dispone di un sensore elettromeccanico che segnala il fine - carta. Ciò significa che se non è presente un foglio di carta al suo interno, l'eventuale procedura di stampa viene interrotta, sia per impedire che la testina scriva sul rullo, sia per evitare la perdita di dati.

Quando la stampante è alimentata con un modulo continuo, il sensore è costantemente abbassato e la stampante viene "vista", dal computer, sempre in grado di ricevere, e stampare, dati.

Quando, invece, si utilizzano fogli singoli, inevitabilmente capita di togliere un foglio per inserirne un altro. In tale intervallo di tempo il sensore "scatta" e pone la stampante in stato di attesa. Tale stato di interdizione permane anche dopo che il nuovo foglio di carta viene inserito; ciò per consentire all'utente di posizionare correttamente il foglio prima di continuare a stampare. Quando tutto è a posto, è necessario premere il pulsante di via libera (di solito denominato *on line*, e, comunque, indicato nel libretto di istruzioni della periferica) per fare in modo che la stampa ricominci.

Il sensore, ottico o meccanico, lo puoi individuare togliendo il foglio di carta e facendo scorrere un biglietto da visita lungo il rullo (nella parte posteriore): quando sentirai un click, oppure quando vedrai la lucetta di *on line* riaccendersi (si era spenta a causa della mancanza di carta), avrai trovato la posizione fisica del sensore.

Tornando al problema, per stampare più fogli singoli con Easy Script, è necessario sostituire il foglio, premere il pulsante di on line (per ripristinare il collegamento computer - stampante) e premere, infine, il tasto C (contiuna), per vedere apparire la stampa su carta.

AMIGA HARD DISK

□ Esiste un hard disk per Amiga? Che cosa è lo *streamer*?
(Francesco Baldacci - Livorno)

• Di recente è stato presentato il modello Amiga 2000HD che dispone, al suo interno, della periferica citata. E' molto probabile che il mercato verrà invaso da molti modelli di questo insostituibile accessorio che, in verità, avrebbe dovuto essere presentato fin dall'inizio, dal momento che Amiga viene definito come un computer professionale (e che professione svolgi, se non puoi usare un hard disk?).

Non appena ci capiterà di averlo tra le mani (e di usarlo in modo intensivo) pubblicheremo le nostre impressioni d'uso, soprattutto per ciò che riguarda la sua invulnerabilità ai virus.

Lo *streamer* è una memoria di massa a nastro. Consente di riversarvi l'intero contenuto di un hard disk e, in seguito, di ricaricarlo. E' indispensabile quando si vogliono effettuare interventi "pericolosi" su h/d, per rendere due computer rigorosamente identici (per ciò che riguarda il software installato) o per ordinaria manutenzione. E' possibile effettuare copie fisiche di h/d servendosi di comuni floppy disk, ma il numero di dischetti occorrenti, e soprattutto il tempo necessario a "scaricarvi" un intero h/d, scoraggia tali espedienti ed obbliga ad usare l'accessorio citato.

CORSI

□ Posseggo un Amiga e vorrei seguire un corso per raggiungere una buona conoscenza della *computer graphics*, volta prevalentemente all'aspetto artistico / creativo. Che cosa mi consigliate?
(Mariano Brandoli - Bastiglia)

• Che io sappia, non vi sono corsi del genere, se non quelli specifici di aziende che, vendendo stazioni grafiche professionali (per studi pubblicitari, emittenti televisive e così via) offrono, oltre all'hardware, anche un servizio di "istruzione".

Ritengo, tuttavia, che se si ha una conoscenza delle tecniche grafiche (disegno, pittura) queste potranno essere applicate con una certa disinvoltura anche con Amiga, ovviamente corredato da uno dei tantissimi tool grafici disponibili.

Nei tempi antichi l'uomo aveva a disposizione solo pietre levigate, sufficientemente ampie per scolpirvi i profili, magari "stilizzati", di animali, uomini e cose. Con il progredire della tecnologia, furono scoperti altri metodi per produrre immagini (vernici, tele) e riprodurle in modo più (litografie) o meno preciso (fotografia, videoregistrazione).

Il computer non è altro che uno strumento, certamente più versatile e creativo di altri strumenti del passato. E, come un qualsiasi strumento, per funzionare correttamente deve essere ben conosciuto da chi lo adopera.

Rimane, semmai, il problema di fondo: essere in grado di produrre qualcosa di più di semplici scarabocchi; essere, insomma, un "artista", o per lo meno una promessa in questo campo.

E con Amiga ciò è decisamente facile.

CON IL C/16 NON SI PUO'

□ Come posso controllare, con il mio C/16, semplici dispositivi elettronici via hardware?
(Mirko Gubbini - Foligno)

• Un qualsiasi computer è prevalentemente costituito da circuiti integrati interconnessi tra loro. In alcuni casi troviamo "traccia" di tali collegamenti nei connettori che si trovano disseminati all'esterno del computer (porte joy, connettore seriale, video e per il registratore), ed al suo interno (connettore della tastiera).

Il C/64 dispone della famosa User Port che permette di gestire otto linee di dati in ingresso o in uscita. Il C/16 non offre tale opportunità e la gestione di apparecchiature esterne risulta quasi impossibile.

Perchè uso il termine quasi? Perchè, in teoria, un vero esperto potrebbe essere in grado di effettuare saldature, più o meno azzardate, in punti precisi della piastra madre ed "estrarre" un bus da collegare ad un

comune connettore. Oppure, sempre pensando alle virtù di un vero esperto, si potrebbe pensare di usare la porta del registratore o del drive che, opportunamente collegate ad un decodificatore (tutto da progettare, costruire e sperimentare) possa offrire qualche bit programmabile in ingresso o in uscita.

L'impresa, come intuitivo, è decisamente ardua ed in ogni caso del tutto antieconomica: costa molto di meno, credimi, cambiar computer.

STAMPA DIFETTOSA

□ Esaminando i fogli di carta stampati con la mia stampante si può notare che i caratteri vengono riprodotti male. Vale la pena ripararla?
(Livio Ariboli - Arezzo)

• Le sbavature che appaiono sistematicamente sono dovute con molta probabilità (come tu stesso hai intuito) al difettoso funzionamento di due aghi.

Rimane quindi il problema: riparare la stampante Mps-802? Forse è meglio utilizzarla così com'è finchè i caratteri risultano ancora leggibili (non sono poi così male). Dopo la definitiva rottura provvedere alla sostituzione: non della testina, ma della stampante!

PROGRAMMI INTROVABILI

□ Non riesco a trovare il Macro Assembler Commodore (o il famoso Logo) nemmeno presso i Commodore Point. A chi è possibile chiederlo?
(da troppe telefonate e lettere)

• Ai pirati, che, almeno in questo caso, effettuano un servizio decisamente migliore della Casa Madre.

LOCAZIONI NON USATE

□ Esaminando la mappa della memoria del C/64 si nota che numerose locazioni, tra cui alcune in pagina zero, risultano non utilizzate. Perchè i progettisti non le hanno usate?
(Roberto Bianucci - Nodica)

• Il microprocessore 6510 dispone di numerose istruzioni che, riferendosi ai primi 256 byte dell'area Ram, permettono elaborazioni rapidissime. E' ovvio che i progettisti hanno sfruttato fino in fondo tale caratteristica per allocare puntatori, vettori ed altri "elementi" che consentissero risparmi di tempo considerevoli.

AIUTATECI A SERVIRVI MEGLIO

Spesso alcuni lettori, che dichiarano di possedere numeri arretrati del nostro periodico, porgono quesiti le cui risposte sono già state esplicitamente pubblicate (in occasione di risposte ad analoghe domande) oppure sono contenute in articoli presenti nei fascicoli in loro possesso.

Per evitare di ripetere argomenti già trattati, pertanto, ricordate di indicare sempre, nelle lettere che inviate, i numeri dei fascicoli in vostro possesso: potremmo infatti indicarvi, se esistono, gli articoli che, in un modo o nell'altro, chiariscono gli argomenti richiesti.

Si ricorda ai lettori che non ci è possibile rispondere privatamente, nemmeno se si acclude l'affrancatura per la risposta.

Per accelerare il servizio, ricordate di indicare sempre la data di spedizione dal momento che questa costituisce diritto di precedenza.

Coloro che non vedessero citato il loro nome nella presente rubrica, possono trovare risposta ai loro quesiti in appositi articoli, apparsi nel frattempo, o di prossima pubblicazione.

Non tutti i byte sono però necessari per svolgere gli innumerevoli compiti del computer e, di conseguenza, alcuni byte non vengono utilizzati nè dal Sistema Operativo nè dall'interprete Basic e rimangono, quindi, a disposizione dell'utente che può usarli come più gli aggrada.

SCROLLING EDITOR

□ Perchè avete affermato che il mio programma "Scrolling Editor" (inviatovi su disco) non è utile?
(S. B. - Codigoro)

• Ne approfitto per ricordare a tutti i lettori che i programmi, per avere una minima speranza di esser pubblicati, devono:
1- Essere inviati su disco.
2- Se in linguaggio macchina, devono essere inviati, in Basic, nel formato Read... Data in modo da consentirne la facile digitazione da parte dei lettori, escludendo difficoltà derivanti da uso improprio di assemblatori vari.
3- Se in l.m. devono esser corredati da disassemblato *commentato*: i nostri lettori vogliono imparare a programmare.
4- Esser corredati da articolo esplicativo (formato Easy Script) su *disco* (e non su carta) che faccia capire, in modo chiaro e semplice, come utilizzare il programma stesso.

PER C/64, MA NON PER AMIGA

□ Come è possibile che alcune potenzialità, presenti su programmi di word processor per C/64, sono del tutto assenti su pacchetti (ben più potenti) per Amiga?
(Domenico Giusto - Mola di Bari)
(Andrea Basutti - Serravalle)

• I programmi di D.T.P. (Desk Top Publishing) per Amiga sono decisamente completi e potenti. C'è da dire, però, che molti di questi lavorano esclusivamente in grafica hi-res. Ciò significa che la pagina che appare sul video, al momento della stampa, viene considerata come una schermata in alta risoluzione (su cui sono "disegnati" i caratteri) e, come tale, trasferita alla stampante.

Se, quindi, sono presenti caratteri in doppia altezza, mezza larghezza, sottolineati (o altro), non viene inviato alla stampante l'apposito comando di ESCape (in grado di attivare i vari font ed il cui uso è esaurientemente descritto sui manuali originali delle stampanti) ma i caratteri stessi vengono "assimilati" a comuni disegni appartenenti alla pagina da stampare.

I manuali di tali pacchetti di DTP sono, tuttavia, molto complessi. E' quindi probabile che ti siano sfuggite le istruzioni che permettono (se esiste la possibilità) di sfruttare le caratteristiche proprie della stampante.

Con il C/64 ciò non si verifica perchè i programmi di Word Processor lavorano sempre in modo testo (e non in grafica, tranne il w/p di Geos e simili). Di conseguenza gli opportuni caratteri di ESCape vengono inviati e rigorosamente accettati dalla stampante.

NON CARICAMENTI

□ Con il mio drive, a volte, non riesco a caricare un programma "originale",

IN ATTESA DI GIUDIZIO

TRIS
(Antonio Diana - Ceccano)

Il gioco del Tris, inviato in tre versioni (una più sofisticata dell'altra) è una delle classiche trasposizioni, su computer, di giochi di intelligenza. Gli algoritmi inviati (descritti in ben otto pagine fitte di diagrammi di flusso) sono sufficientemente validi, ma ritengo che tu abbia dedicato troppo tempo all'abbellimento del programma stesso.

Avresti potuto, ad esempio, inserire una procedura che, prima di ogni mossa, si incaricasse di esaminare tutte le mosse precedenti (memorizzate volta per volta) in modo da scartare quella che aveva costituito una sicura perdita. Alcuni anni fa fu pubblicato un programma in grado di svolgere una simile funzione e che, ad ogni fine partita, registrava su nastro o disco un archivio di partite che, via via, si arricchiva sempre più.

Ti consiglio, in un prossimo futuro, di approfondire l'aspetto strutturale dei programmi che intendi scrivere, anzichè abbellimenti più o meno validi.

ONE - NINE
(Orazio Ruscica - Modica)

Il gioco matematico che hai inviato ("One - Nine") è molto ben curato esteticamente, ma è descritto in modo sommario e non si riesce bene a capire come fare per farlo funzionare correttamente; ma benedetti ragazzi! Possibile che non vi viene in mente di allegare un esempio pratico su come comportarsi quando il computer ti chiede un numero, che immancabilmente non gli va bene (chissà perchè) e poi ne chiede un altro per sentirsi dire "bravo!" e non sapere per quale motivo!

Il secondo programma inviato sullo stesso dischetto ("Istogrammi") è un po' troppo "rigido" e non permette grandi evoluzioni; è presente un algoritmo per lo sfruttamento massimo dello schermo, ma è assente un'idonea procedura per la visualizzazione dei valori immessi.

Ti consiglio, nei prossimi programmi, di metterti nei panni del potenziale utente e di chiederti quali possano essere le opzioni e le utility che realmente possono interessare. In ogni caso sarebbe opportuno, all'inizio del programma, inserire una richiesta del tipo "Vuoi le istruzioni?" e, in caso affermativo, aprire un file di testo Ascii (precedentemente scritto con un word processor) presente sullo stesso disco. In questo modo si evita di "appesantire" il listato e si offre, con una manciata di righe, un'opportunità davvero comoda al potenziale utilizzatore del programma.

AEROBOMBER
(Ivan Tibaldi)

Il gioco che hai inviato è spaventosamente lungo e non riesco davvero a capire perchè occupi ben 241 blocchi di disco. E' probabile che la posizione degli sprite o di qualche routine l.m. ti abbiano costretto a memorizzare un poderoso segmento di memoria.

Se questo è il caso, però, hai usato una tecnica di programmazione poco efficace. Se, infatti, si ha a che far con un breve programma Basic (che occupa pochi K byte nella parte bassa della memoria) ed una zona Ram che occupa la parte alta della memoria, è possibile risparmiar blocchi (e tempo di caricamento) utilizzando un programma che, con la tecnica di overlay, carichi dapprima il programma l.m. (sintassi ,8,1) ed in seguito il programma Basic vero e proprio.

Per ciò che riguarda il gioco, l'animazione non è molto fluida e l'aereo viene sempre colpito senza un preciso motivo. Il sottofondo sonoro (o meglio, rumoroso) è efficace, ma potevi certamente fare di più, soprattutto in corrispondenza del momento in cui l'aereo viene colpito.

ma, paradossalmente, risolvo il problema operando con una copia dello stesso effettuata con il drive di un mio amico. Come mai?
(Fabio Tiezzi - Grosseto)

• La testina di un qualsiasi drive è in grado di leggere un dischetto anche se i dati ivi contenuti sono registrati con una certa approssimazione. Esiste, quindi, una tolleranza di errore che, di solito, viene sopportata tranquillamente dal sistema operativo del drive. Alcuni file e alcuni drive, tuttavia, possono esser meno "tolleranti" di altri e ne consegue quanto da te lamentato. E' quindi probabile che il dischetto "originale" presentasse difetti accettati dal secondo drive, ma inaccettabili dal tuo. La copia del disco, evidentemente, è in grado di eliminare l'incompatibilità apparente tra il floppy originale ed il tuo drive.

GELATI E PROGRAMMI

Il programma "Compriamoci un gelato" (vedi inserto del numero di luglio / agosto) conteneva un errore ed invitavamo i lettori ad individuarlo.

Si tratta di un errore "logico" nascosto nelle righe 230 - 260. In quest'ultima, infatti, si legge...

```
260 gosub 470: print do$(3,x): w = (2-x)/10
```

...e risulta, tra l'altro, identica alla riga 260. Per disporre del quarto parametro è quindi necessario modificarla come segue:

```
260 gosub 470: print do$(4,x): k = (2-x)/10
```

Il programma sembrava funzionare perfettamente ed alcune contraddizioni dei risultati non potevano esser evidenziate.

Ecco, quindi, un esempio pratico di come sia possibile sbagliare a scrivere programmi, senza accorgersi dell'errore, se non dopo attento e minuzioso esame.

INDIRIZZAMENTI 6502

□ Nella guida di riferimento del programmatore del C/64 è dedicato un intero capitolo ai vari tipi di indirizzamenti possibili, in linguaggio macchina, tra cui quello indiretto e quello indicizzato. Però non mi sembra molto chiaro. Potete spiegarmi di che si tratta?
(Vito Catania - Catania)
(Massimo Cavalleri - Bussolengo)

NEWEL s.r.l.

computer e accessori

UNICA SEDE: VIA MAC MAHON, 75 - 20155 MILANO
Tel. 02/323492 solo per negozio e informazioni relative acquisti in Milano - direttamente in sede
Tel. 02/33000036 per ordinazioni da tutta Italia; Fax 02/33000035 in funzione 24 ore su 24
BBS MODEM 02/3270226 (banca dati) al pomeriggio dopo le 13 fino al mattino successivo
Aperto al pubblico nei giorni feriali dalle 9.00 alle 12.30 e dalle 15.00 alle 19.00
e il sabato dalle 9.30 alle 13.00 e dalle 14.30 alle 18.30 - chiuso il lunedì

DAL 20 GIUGNO AL 20 LUGLIO VENDITA PROMOZIONALE A PREZZI SCONTATISSIMI IN OCCASIONE DEL 10 ANNIVERSARIO NEWEL

AMIGA SHOP - A CASA TUA DIRETTAMENTE 02/33000036 - PREZZI CHIAVI IN MANO

I NOSTRI DISK DRIVE

DISKDRIVE SLIM - Meccanica NEC - beige

sono disponibili:

per Amiga 500 3,5" passante compreso disconnect L. 239.000

per Amiga 500 5,25" 40/80 tracce passante L. 350.000

per Amiga 2000 interno L. 179.000

per C-64 OCC118 L. 239.000

AMIGA MODEM 2400 PAK

Modem dedicato per A500 - A1000 - A2000 esterno 300, 1200, 2400 baud (V21 - 22 - 22 bis). Autodial, autoanswer, Hayes compatibile, completo di software e cavo di connessione al computer (disponibili altre versioni, 300/1200 e 300/1200 - 1200/75 Videotel).
L. 339.000

MINI GEN

MINI-GEN una grande novità per professionisti ed entusiasti, per ottenere sovrapposizioni di animazioni, titoli, messaggi ecc.
Funziona con tutti gli Amiga ed è compatibile con programmi come TV-text, Pro video e molti altri.
Ora la videotitolazione è alla portata di tutti, semplicissimo da usare.
L. 399.000

VIDEON

Basta con i noiosi filtri per i vari passaggi... Ora c'è VIDEON!
Il Videon è un digitalizzatore video a colori dotato di un convertitore PAL-RGB con una banda passante di 15 KHz per ottenere immagini a colori dalle stupefacenti qualità ...
Funziona in risoluzioni di: 320 x 256 - 320 x 512 - 640 x 256 - 640 x 512.
Può essere collegato a una qualsiasi fonte video PAL, ad esmpio videoregistratori, computer, telecamere, televisori, ecc. Il prodotto permette di visualizzare il segnale video collegato all'apparecchio e in più permette la regolazione di luminosità, colore, saturazione, contrasto.
E' corredato di software che permette la manipolazione di immagini IFF HOLD MODIFY da 32 a 4096 colori con tecniche di SURFACE-MAPPING su solidi geometrici.
L. 420.000

ATARI - ST

DRIVE 1Mb L. 290.000

Digitalizzatore video in tempo reale L. 179.000

BOOTSELECTOR

Trasforma il secondo Drive (df1:) in (df0:) evitando così l'eccessiva usura del medesimo, risolve spesso molti problemi di caricamento dovuti alle precarie condizioni del drive interno dopo un uso frequente, semplice da installare (non necessita saldature).
Istr. italiano.
L. 23.000

DISPONIBILI TUTTE LE ULTIME NOVITA' SOFTWARE ORIGINALI

DISCONNECT

Per sconnettere il secondo drive senza dover spegnere il computer, basta agire su un apposito interruttore, recuperando così memoria che spesso necessitano molti programmi, che altrimenti non funzionerebbero.
L. 23.000

ANTIRAM

Questo kit, sconnette tutte le espansioni di memoria su Amiga, sia interne che esterne, risolvendo anche qui i problemi di incompatibilità con il software, semplice installazione.
Istr. italiano.
L. 23.000

OFFERTA!!!

Bootselector + Disconnect + Antiram
L. 59.000

VIRUS DETECTOR PLUS

Utilissimo per i noiosi e nocivi virus che si diffondono facilmente, questo dispositivo hardware è in grado di segnalare con un segnale acustico quando un virus va a scrivere su un disco, evitando così che il virus vi rechi dei danni, molto utile per tutti gli Amiga, si attacca alla porta drive o al drive
L. 35.000

VIRUS DETECTOR PLUS VIRUSKILLER SOFTWARE

L. 45.000

AMIGA MOVIOLA (NOVITA')

Eccezzionale novità, permette di rallentare un gioco fino a 100 a 0, per poter superare tutti gli ostacoli e capire con calma il gioco, molto utile anche per programmi grafici, animazioni, cad, ecc. Puoi variare la velocità di esecuzione, cartuccia completa di istr. italiano.
L. 79.000

TASTIERA MIDI PER AMIGA

YAMAHA + INTERFACCIA MIDI PROF.
L. 299.000

REALTIME GRABBER AMIGA

Digitalizzatore in tempo reale, in b/n per digitalizzare immagini provenienti da una qualsiasi fonte video senza bisogno di avere un fermo immagine, risultati eccezionali a livello fotografico.
Predisposto per lo splitter (vedi sotto).
L. 599.000

AMIGA SPLITTER NEWEL RGB/PAL CONVERTER

Per chi possiede già un digitalizzatore video del tipo Amiga Eye, Amiga Vid, Easy View, Digi View, ecc. Evita il passaggio dei noiosi tre filtri. Lo splitter Newel converte direttamente l'immagine a colori, indispensabile per chi possiede un digitalizzatore in tempo reale in b/n con Newel splitter potrà ottenere risultati straordinari.
L. 285.000

AMIGA EPROM PROGRAMMER

Nuovo programmatore di eprom per Amiga, si collega semplicemente alla porta parallela dell'Amiga e permette di programmare tutte le EPROM dalle 2716 alle 27512 e 27011, il tutto copleto di software di gestione con lettura, scrittura e verifica delle EPROM, molte opzioni come prog. veloce tramite algoritmi, ecc. Semplice da usare completo di istruzioni per l'uso.
L. 229.000

ESPANSIONI DI MEMORIA AMIGA

A501 Espansione originale Commodore che porta a 1 Mb il tuo A500.
L. 319.000

AMIGA PROFEX espansione esterna da 2 Mb, autoconfigurante, swich on/off per A500
L. 1.290.000

AMIGA 1000 RAM, Espansione da 2 Mb per A1000 esterna autoconfigurante
L. 1.290.000

AMIGA 2000 RAM, Espansione interna da 2 Mb originale Commodore.
L. Telefonare

Tutte le espansioni sono fornite complete di chip ram e garanzia 12 mesi!!!

AMIGA BOX TRANSFORMER

Il famoso box di espansione "Big Blue" ora disponibile nella nuova versione per trasformare un Amiga 500/1000 in Amiga 2000, si può così risolvere il problema delle espansioni di memoria e delle schede XT & AT per l'emulazione MS-DOS, questo cabinet è predisposto già per 2 floppy da 3,5", 1 floppy da 5,25", hard disk + 3 slot in Amigados, 3 slot IBM XT compatibili, 3 slot IBM AT compatibili, 1 slot per scheda velocizzatrice 68020/68881. Potrete quindi utilizzare tutte le periferiche dell'Amiga 2000 (dai un tocco di professionalità al tuo Amiga)
L. 399.000

AMIGA MOUSE

Finalmente disponibile il mouse di ricambio originale Commodore, dedicato per Amiga 500/1000/2000
L. 89.000

AMIGA FAX

Straordinario FAX per Amiga, permette di inviare e di ricevere segnali fax, cartine, ecc. Completo di hardware di gestione, disco & manuale in italiano, l'installazione e l'uso sono di una semplicità estrema.
Lit. 199.000

CMI ACCELLERATOR BOARD

Scheda accelleratrice per Amiga 500/1000/2000 raddoppia la velocità del tuo Amiga portandola a 16 Mhz, molto utile per chi usa programmi grafici con VIDEOSCAPE, SCULPT, VIDEO EFFECT, PRO VIDEO e molti altri, predisposto per coprocessore matematico 68881.
Metti il turbo al tuo Amiga!
L. 499.000

CMI COPROCESSOR - 68881

L. 299.000

HARD DISK ESTERNO 20 Mb per Amiga 500

in offerta L. 990.000

HARD DISK AMIGA CARD 20 Mb per Amiga 2000

L. 990.000

GVP HARD DISK con Autoboot per Amiga 500 (Fast File System DNA)

L. call.

GVP HARD DISK con Autoboot 20 Mb con controller (Fast File System DNA)

L. 1.390.000

GVP HARD DISK con Autoboot 40 Mb con controller

(Fast File System DNA)
L. 1.690.000

GVP HARD DISK con Autoboot Hard Quantum 45 Mb 11 ms. Espansione 2 Mb (Prodrive)

L. 2.890.000

HARD DISK per Amiga 2000 (Scheda) (con scheda XT-AT) partizionabili:

20 Mb	619.000
32 Mb	759.000
40 Mb	939.000

AMIGA ACCESSORI IN OFFERTA

Drive 3,5" esterno per Amiga Slimline passante L. 229.000

Drive 3,5" come sopra più disconnect incorporato L. 239.000

Drive 3,5" interno per A2000 NEC (Con viti ecc.) L. 179.000

Drive 5,25" esterno novità (Amigados + MS-DOS) L. 329.000

Drive 5,25" OC/118 Drive per C64 o Amiga + Emulator L. 249.000

Disponibili anche i nuovi AMIGA DRIVE Newel con display Trak

Interfaccia Midi Professionale per AMIGA L. 79.000

Pro Sound Designer GOLD Vers. Dig. Audiostereo!!! L. 169.000

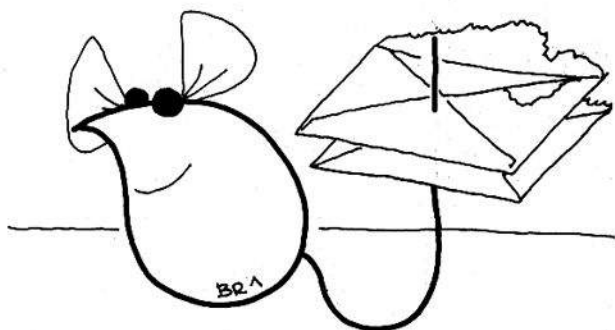
Scheda Janus XT per Amiga 2000 per la comp. MS-DOS L. 980.000

Scheda Janus AT per Amiga 2000 per la comp. MS-DOS L. 1.750.000

Le schede sono complete di Disk Drive 5,25" e manuali + Software

KICKSTART 1.3 ROM

Il nuovo sistema operativo dell'Amiga ora in ROM applicabile facilmente su A500 e A2000 senza saldature e senza perdere il vecchio 1.2, disponibile anche l'inverso per chi possiede 1.3 e vuole 1.2, con interrutore per selezionario. NOVITA' KICKSTART in ROM + Orologio per A1000 esterno (New!!!)
L. 119.000

SPEDIZIONI CONTRASSEGNO IN TUTTA ITALIA CON POSTA O CORRIERE

• Sull'inserto del N. 58 di C.C.C. è riportato un articolo ("Un antilist a prova di Hacker") che affronta in modo pratico ed esauriente il complesso argomento.

L'articolo citato fa parte della serie dedicata, appunto, all'Assembly, che il nostro validissimo Domenico Pavone ha affrontato, sistematicamente, fin dal N. 51, in cui comparve il primo della serie.

Come è facile immaginare, l'argomento è troppo complesso per essere affrontato in queste pagine e non posso quindi rispondere adeguatamente.

Tieni presente, ancora, che il fascicolo speciale "Commodore, linguaggio macchina e routine grafiche" è suddiviso in due parti, la prima delle quali rappresenta un vero e proprio corso sul linguaggio macchina del C/64. Tra gli argomenti trattati, ovviamente, figurano tutti gli indirizzamenti possibili con il 6502.

Ti assicuro, tuttavia, che prossimamente torneremo ad occuparci di tecniche legate alle istruzioni che ti sembrano, per ora, così poco chiare.

Tieni presente che, in ogni caso, le tecniche di indirizzamento sono grosso modo simili (almeno concettualmente) in tutti i microprocessori. Imparare ad usarle bene con il C/64 permetterà di risparmiare tantissimo tempo quando, passando ad un sistema "superiore" (Amiga, Ms-Dos), deciderai di studiare l'Assembly specifico.

ARCHIVI SU NASTRO

□ Vorrei informazioni su programmi che, su nastro, consentano di gestire un qualsiasi tipo di archivio.
(Antonio Russo - Collepasso)

• Anche se esistessero programmi generici del tipo descritto, non ne consiglierei l'uso perchè un computer deve far risparmiare tempo e non perderlo.

Un archivio su nastro (che, quindi, non può disporre della versatilità dei file relativi, esclusiva caratteristica dei drive) è costretto ad operare con i soli dati presenti in memoria Ram. Dal momento che un C/64 dispone di poche migliaia di byte, è impossibile gestire grandi quantità di dati. E siccome la quantità di dati può solo esser modesta, risulta, in definitiva, più semplice lavorare con carta e penna, rinunciando all'uso del computer.

Archivi di dati su nastro, tuttavia, possono esser considerati efficienti se progettati esclusivamente per specifiche applicazioni. Solo in questo caso, infatti, è possibile scrivere un programma che, ottimizzato al massimo, consente non solo di trattare un maggior numero di dati, ma anche di sveltire procedure di ricerca complesse. Nel caso di programmi "generici", al contrario, la richiesta versatilità impedisce lo sviluppo di procedure ottimali.

IL MIGLIORE

□ Posseggo l'Amiga ed ho avuto una discussione con un sostenitore del sistema Ms-Dos. Qual è il miglior sistema per effettuare disegni tecnici e D.T.P?
(Pierluigi Covelli - Firenze)

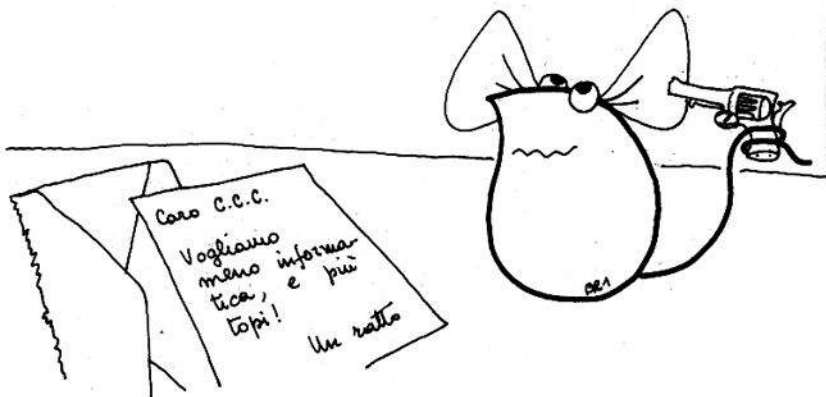
• Siamo alle solite: è meglio un Ms-Dos oppure l'Amiga? Sembra una domanda del tipo "E' più forte l'Inter o il Milan?"; prova a chiederlo prima ad un Interista e poi ad un Milanista; e non chiamare in causa anche Juventini, Laziali, Romanisti perchè, altrimenti, non la finiresti più.

Ogni sistema presenta pregi e difetti. Se un sistema offrisse soltanto difetti verrebbe subito abbandonato al suo triste destino. E non mancano certo esempi di clamorosi fallimenti commerciali relativi a macchine del tutto inadeguate alle esigenze del mercato.

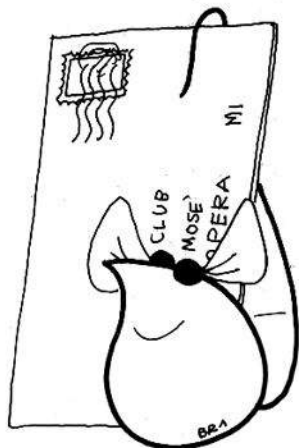
Una risposta "quasi" obiettiva si potrebbe ottenere soltanto da quei professionisti che, avendo la possibilità di sperimentare *entrambi* i sistemi per *lungo* tempo, riescano ad individuare le particolarità che li pongono in evidenza. Ed in ogni caso, credimi, vi saranno sempre applicazioni (o casi particolari) in cui un sistema si comporta meglio dell'altro. O peggio.

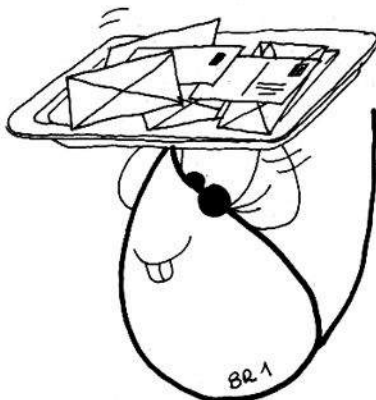
IN CANTIERE

• Alcuni lettori (tra cui G. Caturano, Mario Brezzolari, Maurizio Fuschetto) propongono la realizzazione di piccoli accessori hardware. Ricordiamo che se tali apparecchiature sono realmente semplici da realizzare, saremo felici di pubblicare il lavoro di detti lettori. Contattateci telefonicamente *dopo* aver realizzato l'accessorio, in modo da concordare la pubblicazione dell'articolo relativo.

Il presente avviso è valido anche per i lettori che vogliono proporre software (tra cui Ugo Spezza, Diego Zuccolo, Giuseppe Miduri, Andrea Fasce)

risposte rapide

AMIGA MONOCROMATICO
(Luca Bertin - Padova)

Amiga è un computer con uscita a colori e suono stereo. Il collegamento con un monitor monocromatico è certamente possibile (è presente uno specifico connettore sul retro); ma che vantaggio avresti a risparmiare poche(!) centinaia di migliaia di lire?

COMANDO USR
(Valerio Capello - Nuoro)

Il comando USR è stato abbondantemente descritto in un apposito articolo ("Affintà elettive tra Basic e Linguaggio Macchina") pubblicato sul N. 50 di Commodore Computer Club.

GIA' FATTO
(Alessandro Banchero - Selargius)

Le Poke che suggerisci per il C/128 sono state abbondantemente descritte in precedenti fascicoli della rivista e non posso ripubblicarle.

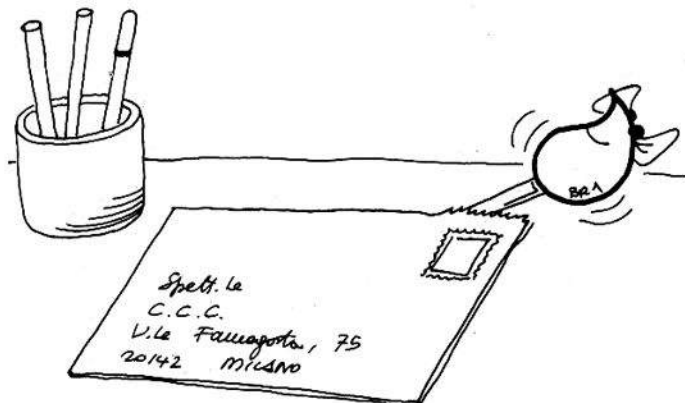
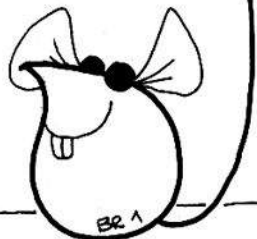
MENO 26627
(Orazio Stracuzzi - Messina)

La risposta che ottieni dal tuo C/64 appena acceso, impartendo il comando Print Fre(0) è negativa (-26627) per motivi più volte citati in passato. Per conoscere esattamente il numero di byte disponibili è necessario digitare Print 65536 + Fre (0). Il numeto 65536 è il risultato di 2 elevato alla 16ma potenza.

COMANDI EASY SCRIPT
(Giuseppe Sciacca - Aci Catena)

L'intero inserto del N. 41 era dedicato all'uso di Easy Script e alla descrizione di quasi tutti i comandi del potente word processor per C/64. Per avere l'intero elenco è necessario procurarsi il manuale originale che consta di parecchie decine di pagine, perfettamente comprensibili (e in italiano!).

SOLO SU DISCO
(Simone Pistolesi - Siena)
(Alfredo Paganelli - Modena)
(Giancarlo Voci - Mantova)

Possiamo prendere in esame (e cercare di rintracciare eventuali errori) i programmi che pervengono in redazione su *disco e non su carta.*

MEGLIO ENTRAMBI
(Piero De Giorgio - Taranto)

Non ti consiglio di vendere il tuo C/64; tanto più se, come asserisci, anche dopo l'acquisto di Amiga prevedi di continuare ad usarlo.

PURTROPPO E' GUASTO
(Lino Fiorillo - Sant'Antimo)

Se il tuo C/128 non riconosce mai la pressione di determinati tasti, non c'è alternativa: devi portare il tuo computer presso un centro di assistenza.

DA 500 A 2000 IL PASSO E' BREVE

La maggior parte dei possessori di Amiga utilizza un 500 sognando un 2000; è giunto il momento di svegliarsi

di **Michele Maggi**

Tra un Amiga 500 con l'espansione di memoria ed un Amiga 2000 non esistono, inizialmente, grandi differenze funzionali, a parte le dimensioni ed il costo iniziale: il grosso vantaggio 2000 è la maggiore espandibilità.

I cataloghi della Commodore, e di molti costruttori indipendenti, sono affollati di schede di ogni tipo, adatte alla collocazione nel capace interno del "cabinet" del 2000.

Queste schede di espansione vanno inserite negli appositi slot (ne sono presenti 7) che consentono di montare schede Amiga ed IBM, e che permettono, al possessore dell'A-2000, di potenziare in misura considerevole le caratteristiche della macchina.

E' oramai chiaro che il mercato appare "maturo" nei confronti dell'A-2000 e si sta evolvendo necessariamente verso una sempre maggiore produzione dedicata a questa macchina.

E IL 500?

Di fronte a tali considerazioni, l'utente che ha scelto di acquistare l'Amiga 500 rimane necessariamente contrariato: ovviamente chi desidera acquistare il 2000, senza pensare di aggiungere qualche espansione, potrebbe tranquillamente rivolgersi al 500, mentre non sempre chi acquista il 500 non si trova ad avere bisogno (o semplicemente voglia) di espanderlo.

La scelta iniziale riguardo al computer da acquistare e, purtroppo, troppo spesso causata dalla dittatura del vile denaro, ma se crescono le esigenze (e se il portafoglio lo permette) può presentarsi la necessità di un "upgrade" verso l'Amiga più grande.

Purtroppo la spesa è, in questo caso, piuttosto elevata: la somma destinata al modello 2000, sommata a quella delle espansioni, è capace di infliggere duri colpi anche ai portafogli più forniti.

Fortunatamente si è recentemente proposta una geniale alternativa, che sfrutta la capacità dell'Amiga 500 di trasferire tutti i segnali dei propri chip sul bus laterale ad 86 pin: è stata quindi approntata una scatola magica che permette di accrescere l'espandibilità del computer avvicinandola a quella del 2000.

Tale possibilità è inoltre offerta anche ai possessori di Amiga 1000, computer per il quale è già stata approntata una versione dedicata, modificata nelle dimensioni e nella posizione del connettore del bus, che sul modello 1000 è sulla destra.

Il nome dell'accessorio è Zorro Big Blue perchè permette di inserire fino a tre schede Amiga/IBM ("Zorro è il nome degli slot standard Amiga, e Big Blue è il "soprannome" dell'IBM) in un contenitore che può ospitare comodamente, oltre alle schede in formato lungo, anche due drive da 3.5 pollici, uno da 5 e un quarto (per la compatibilità IBM), un hard disk da 3.5 ed un alimentatore da 40 Watt per dare vita al tutto.

ALL'ESTERNO...

Zorro Big Blue per l'A-500 è un box metallico delle dimensioni di 46 X 37 cm. verniciato in beige Amiga.

Sulla faccia superiore (che funge da coperchio) è presente una feritoia dalla quale fuoriesce il connettore della scheda Zorro Big Blue Bus che andrà a collegarsi con il bus dell'A-500, che di conseguenza dovrà essere posizionato al di sopra di Zorro: sul lato posteriore sono presenti i fori per l'in-

QUATTRO DOMANDE AI MEDIA D'AGENZIA E D'AZIENDA

1	Pianificate il mezzo "radio privata" o avete in programma di farlo?	SI	NO
2	Conoscete i palinsesti dei programmi di tutte le "radio private" per una precisa scelta del target group?	SI	NO
3	Avete la garanzia di aver ottenuto o di poter ottenere il miglior trattamento commerciale e creativo nella pianificazione di "radio private"?	SI	NO
4	Avete avuto la sicurezza e la garanzia ufficialmente certificata, da un istituto riconosciuto dalla categoria, della corretta avvenuta messa in onda dei comunicati pianificati sulle "radio private"?	SI	NO

- Se a queste 4 domande avete risposto sempre SI, avete già avuto rapporti con Egimedia e TIR Top Italia Radio.
- Se a queste 4 domande non avete risposto sempre SI, Egimedia e la AGB ITALIA, per TIR Top Italia Radio, sono l'**unico** interlocutore oggi che Vi farà rispondere SI a tutte le 4 domande ed altre ancora.

PER UN MIGLIORE SERVIZIO AI VOSTRI CLIENTI E ALLE VOSTRE AZIENDE

EGIMEDIA SRL - VIA DELLA SPIGA 1 - 20121 MILANO - TEL. 02/79.85.31 - 79.45.92

terruttore ed il cavo di alimentazione, quello per il passaggio dei cavi dei drive aggiuntivi e, sulla destra, le tre guide per le schede tipiche del mondo Amiga 2000 ed Ms-Dos.

Sul frontale del cabinet sono presenti le sedi per i due drive da 3.5 e per quello da 5 1/4, più il led di alimentazione e quello per il funzionamento dell'hard disk: se il relativo foro non viene impiegato, le sedi dei drive rimangono coperte da mascherine di plastica; su tutto il contenitore sono praticate delle feritoie per il raffreddamento delle schede.

E ALL'INTERNO...

All'interno della Zorro (consideriamola per ora nuda, cioè priva di espansioni) troviamo i telaietti atti a sorreggere i drive; quindi, verso il fondo, i fori per fissare le colonnine che sosterranno l'alimentatore; sulla sinistra, invece, si posiziona la piastra madre (la Zorro Big Blue Bus) che contiene i componenti passivi che permettono di conservare i livelli dei segnali ed i connettori (gli slot) dove verranno inserite le schede di espansione.

Il tutto è costruito ed assemblato con notevole cura, e vanta una notevole robustezza (data anche la costruzione metallica); le piste della scheda sono pulite e lasciano intendere una accurata progettazione ed ottimizzazione, i connettori sono sicuri e resistenti.

Le schede andranno inserite in posizione orizzontale negli slot, che come già specificato accettano tanto le espansioni autoconfiguranti di Amiga (negli slot a 100 pin) quanto quelle IBM nello standard XT o AT: dato che gli slot sono allineati su tre file, ecco spiegato il perchè non è possibile inserire più di tre schede per volta.

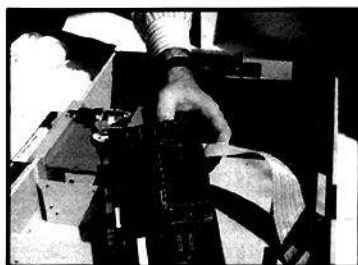
Una considerazione importantissima: il connettore da 86 pin di Amiga, che viene occupato da Zorro, è replicato per consentire l'inserimento di eventuali schede acceleratrici.

COME FUNZIONA

Abbiamo provato la Zorro utilizzando una Bridgeboard AT ed un drive da 5 e 1/4, caricando l'Ms-Dos dato in dotazione alla scheda stessa, e che riporta la scritta "Commodore Ms-Dos for Amiga 2000"; inoltre abbiamo testato la funzionalità del prodotto con una espansione di memoria A 2058 da 2 Megabytes (espandibili ad 8) e con un hard disk dedicato ad Amiga (un Miniscribe da 3.5, con 20 Mb di capacità e tempo di accesso di 40 millisecondi).

Le nostre impressioni sono state positive: la macchina ha funzionato alla perfezione, consentendoci operazioni alle quali l'Amiga 500 non è abituato, e facendoci veramente credere di avere a che fare con un 2000: con il Kickstart 1.3 l'hard disk è andato perfettamente in autboot, la Bridgeboard ha funzionato al meglio e il ramtest della Commodore è stato brillantemente superato dall'espansione di memoria.

L'unico appunto che possiamo rivolgere alla Zorro Big Blue è dovuto al fatto che, essendo costretti per ragioni di segnale a mantenere molto corta la distanza tra il bus e gli slot, l'Amiga deve essere posto sopra alla Zorro: l'altezza della tastiera sale notevolmente, rendendo un po' meno agevoli le operazioni di digitazione.

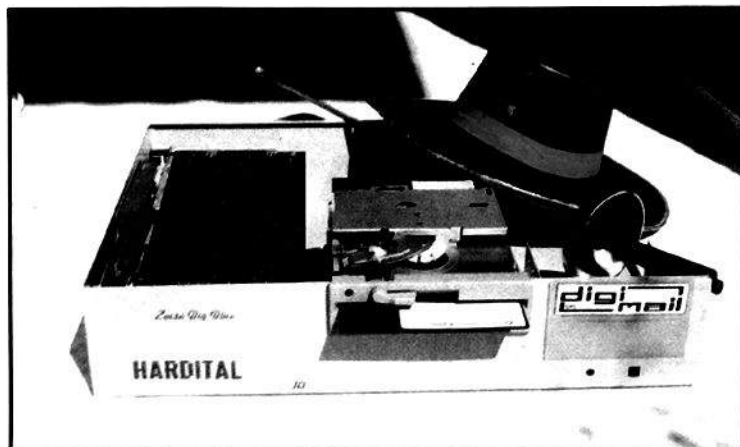
Ma, a conti fatti, questo piccolo difetto della Zorro passa in secondo piano quando ne viene considerato il prezzo, che è di

sole 150.000 Lire (IVA compresa) per lo chassis metallico, già utile di per sè per l'inserimento dei drive esterni, alla quale vanno eventualmente aggiunte le 180.000 necessarie per l'acquisto dello Zorro Big Blue Bus.

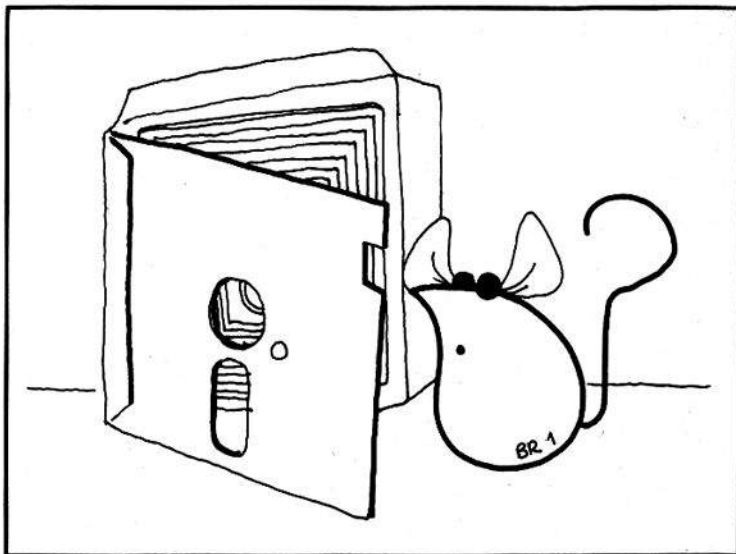
L'alimentatore (necessario solo se si monta un hard disk) costa 99.000 Lire (IVA compresa), mentre i prezzi delle schede e dei disk drive (hard e floppy) variano e dipendono dal tipo di scheda desiderato.

A CHI E PERCHE'

I primi utenti ideali della Zorro Big Blue potrebbero essere i possessori di Amiga 1000, lasciati un po' in disparte dalla Commodore che non ha più importato (nè sviluppato) espansioni dedicate a questa macchina.

In questo modo essi potrebbero riempire il buco esistente tra il primo Amiga ed il suo successore professionale A 2000; il tutto senza alcuno svantaggio estetico o funzionale, anzi, la sistemazione dei drive inseriti nella Zorro porta certamente un tocco di praticità, anche se verrà necessariamente alzata la posizione del monitor.

Anche gli utenti di Amiga 500 che volessero sfruttare al meglio le possibilità offerte dell'hardware (senza limitarsi al gioco...) saranno certamente interessati alle caratteristiche proprie della scatola magica, che consentirà loro un notevole salto qualitativo con una spesa limitata (che ovviamente, però, ne implica altre, dato che non avrebbe senso comprare la Zorro e non espanderla in seguito) che vale certamente la pena affrontare.

Per ulteriori informazioni:

Digimail S.r.l.
Via Coronelli, 10
20146 - Milano
Tel. 02-426559

FIRMATE I VOSTRI DISCHETTI

Una procedura utile per personalizzare i vostri dischi, ma soprattutto per imparare qualche trucchetto in più

di **Massimiliano Alessandri**

Sicuramente vi sarà capitato, per ragioni di Hackering o semplicemente per personalizzare un vostro lavoro, di voler inserire un messaggio nella Startup-Sequence di un disco.

E certamente vi sarete scontrati con le numerose difficoltà per realizzare una utility specifica. Il programma di queste pagine, *Install message*, è un Batch File che risolve per voi il problema, visto che consente di effettuare l'operazione in modo pratico e piacevole: qui di seguito sono riportate le sue caratteristiche.

• La creazione di un "messaggio di default", che conserverete sul vostro disco-programma, e che verrà copiato (sempre che lo vogliate) senza modifiche sul disco da "personalizzare".

• La protezione di tutti i files interessati all'operazione, allo scopo di rendere ardua la rimozione del vostro messaggio da parte dei meno abili.

• La possibilità di effettuare un *Install* sul disco personalizzato.

• La creazione di una nota (ingannatoria) sul file del messaggio per scoraggiare la sua rimozione (*Erase this file and you will corrupt main program data !*).

COME INSTALLARE IL PROGRAMMA

Effettuate una copia del vostro Workbench 1.2 su di un disco vergine, quindi digitate i due listati (*Starter* ed *Install message*) e salvateli entrambi nella directory S (usate preferibilmente il MicroEMACS). Comunque potrete utilizzare anche un qualsiasi altro disco: è sufficiente che la sua directory C contenga tutti i comandi AmigaDOS utilizzati dal programma. Fatto ciò, il Batch File pronto per l'uso.

Inserite il disco contenente il Batch File, abilitato alla scrittura, in un qualsiasi drive. Entrate in ambiente CLI e digitate:

EXECUTE STARTER

Appariranno alcuni messaggi relativi al caricamento, poi tutti i comandi AmigaDOS verranno copiati nella RAM Disk.

Terminata questa procedura (che non si ripeterà più fino ad un successivo Reset grazie ad un apposito accorgimento) il computer controllerà l'esistenza del messaggio di default sul disco-programma, ed in caso di esito negativo, vi porrà, dopo una pausa di pochi secondi, in modo *Edit*.

Utilizzate i normali comandi che usereste con il comando ED (vedi appendice B pagine 3 - 5 del manuale dell'Amiga 500),

poi premete ESC seguito da X per salvare permanentemente il messaggio. A questo punto il computer chiederà se volete effettuare modifiche alla copia in RAM del messaggio di default (che è quella che poi, in definitiva, andrà sul disco da personalizzare); premete Y seguito da Return (se lo volete), altrimenti premete solo Return.

Se avete scelto di modificare il messaggio, usate sempre i comandi di ED, come spiegato sopra. Ora dovrete inserire il disco nel quale volete installare il messaggio nel drive interno (importante!) e premere Return.

Attendete almeno 5 secondi dopo l'inserimento di questo disco, prima di premere nuovamente Return, per evitare errori! In pochi secondi il comando Type ed il messaggio verranno scritti nelle corrette directory, automaticamente create dal computer in caso di mancanza delle stesse, e poi verrà chiesto se desiderate INSTALLare il disco. Rispondete come descritto sopra. Questa operazione può essere utile per rimuovere introduzioni già realizzate da altri nel BootBlock, oppure per eliminare un eventuale virus.

E' giunto ora il momento di inserire il fatidico comando *Type S/Message* nella Startup-Sequence (questa operazione non poteva essere automatizzata in AmigaDOS): dopo una pausa di cinque secondi entrerete in modo EDIT (usate sempre i comandi di ED) e potrete inserire il comando nella posizione desiderata (è consigliabile inserirlo nelle primissime linee).

Verrà quindi chiesto se volete proteggere (con il comando *Protect*) tutti i files interessati dalla cancellazione; rispondete come alle precedenti domande, poi alla apposita richiesta di sistema, reinserite il disco del programma in un qualsiasi drive. Da notare come la nota ingannatoria venga comunque inserita nel file del messaggio. Il computer chiederà se volete utilizzare nuovamente il programma: se sceglierete di farlo, potrete risparmiarvi l'attesa del caricamento dei comandi, poichè sono già in RAM.

Ricomincierete così l'operazione dalla modifica del messaggio in RAM. Rispondendo NO, ovviamente, tornerete in ambiente CLI; per liberare la RAM, comunque, sarà meglio RESETtare il computer.

Un'ultima nota: per cambiare il messaggio di default potrete modificare il file Message, posto nella directory S del disco-programma, con un qualsiasi editor.

ALTRI USI

Un altro uso a cui si presta Install Message è quello di trasformare un banale dischetto dati in uno con auto-boot (grazie alla possibilità di INSTALLare il disco e di creare una Startup-Sequence), con un messaggio automaticamente installato e visualizzato; altro uso può essere ancora quello di fare in modo che appaia il vostro nome su di un programma che intendete spedire in prova ad un possibile acquirente.

```
; STARTER for INSTALL MESSAGE.  1989 by Massimiliano Alessandri.

Echo "*NLoading 'InstallMessage' ..."
Echo " 1989 by Massimiliano Alessandri"
IF NOT EXISTS RAM:InstallMessage
Copy SYS:s/InstallMessage to RAM:
EndIF
Execute RAM:InstallMessage
Ask "*NOK. Un'altro uso (y/n) ?"
IF WARN
Execute Starter
Quit
EndIF
Echo "*NINSTALL MESSAGE terminato."
Echo "(se vuoi liberare la RAM Disk resetta il computer)"
```

Mk V' non solo è la miglior cartridge per effettuare copie di sicurezza del proprio software, ma è anche il più efficace velocizzatore nastro/disco e la più versatile cartuccia di utility esistente.

La sua peculiarità più innovativa è data dal suo microprocessore interno, appositamente studiato per sovrapporsi a quello del computer, ed assolutamente invisibile al sistema.

Ad esempio, mentre le altre cartucce si fermano ad un banale "Sprite Killer" per facilitare i giochi, Mk V' è in grado di trovare automaticamente le "Poke" necessarie per le vite infinite di qualsiasi programma presente e futuro, senza attendere che siano pubblicate dalle riviste o che qualche smanettone studi il programma. Ora anche voi potete produrre giochi "trainer", senza alcuna conoscenza di linguaggio macchina! Inoltre sprotegge e porta da nastro a disco (e viceversa) qualsiasi programma protetto, anche in multiload (con i parametri in dotazione); può trasferire molti programmi e files dal formato 5"1/4 al nuovo 1581 da 3"1/2; velocizza il nastro 5-6 volte oppure 8-10 volte, con velocità selezionabile; velocizza il disk drive come se fosse parallelo (2 velocità: 202 blocchi in 9 secondi oppure in 6!), ed è sempre efficace, anche con i programmi che disabilitano i fastload normali. Mk V' incorpora un vero e proprio editor di schermo, per poter cambiare più facilmente e velocemente le scritte nelle schermate o nei programmi; funziona da interfaccia parallela, per collegare una qualsiasi stampante standard Centronics al C64/128 ed usarla all'interno di qualsiasi programma, anche grafico; stampa o di salva in qualsiasi momento la schermata o gli sprites di un gioco, per alterarli a piacimento. Aggiunge nuovi comandi al Basic, monitor L/M e disk, crea serie di immagini in sequenza su nastro, e tantissime altre cose ancora.

Per Commodore 64 e 128 (in modo 64), con qualsiasi registratore o disk drive, originali o compatibili.

ASSICURATI ANCHE TU LA MIGLIOR CARTUCCIA PER C-64/128!

Mk V', manuale in italiano, garanzia 5 anni 99.000
Cavo Centronics per Mk V' 39.000
Enhancement Disk - utilities e parametri speciali 19.000
Graphic Disk, nuovo disco di utility per Mk V con SlideShow di immagini, Sprite Editor Deluxe, Message Maker ad altro ancora 19.000

MODEM

Tutti i modem sono Hayes compatibili e comprendono il software di gestione. I modelli esterni sono dotati di alimentatore proprio e sono compatibili con tutti i computer con porta seriale RS-232; i modelli interni si inseriscono in tutti i PC e compatibili.

Modello	Prezzo
300/1200 interno	189.000
300/1200/2400 interno	319.000
300/1200 esterno	239.000
300/1200/2400 esterno	349.000
300/1200/2400/4800 esterno	849.000
300/1200/2400/4800/9600	1.899.000
300/1200/2400 specifico per PS/2	489.000
300/1200 + Videotel interno	229.000
300/1200 + Videotel esterno	299.000
300/1200/2400 + Videotel esterno	739.000

DIGITALIZZATORI

Framer, il miglior digitalizzatore per Amiga, in tempo reale e con una qualità video stupefacente 1.199.000

Sampler 64, professionale campionatore di suoni per C64/128, non si limita a portare nella memoria del computer i suoni ma li elabora per creare effetti incredibili , compatibile MIDI, con un potentissimo live-sequencer ed editor. Fornito con ComDrum, una batteria elettronica professionale, vi farà ricredere sulle potenzialità musicali del vostro C64 .... 149.000

Digitalizzatore audio per Amiga stereofonico 179.000

Scheda digitalizzatrice video per PC real-time .... telefonare

SHORT-CIRCUIT MAKER

Dispositivo per tutti i computer dotati di porta parallela, crea all'interno del vostro computer uno strano fumo azzurrognolo rendendo inservibile la macchina. Utile per convincere il capufficio o i genitori a cambiare computer con un modello più recente.

In offerta a 29.000 rimborsabili all'acquisto del computer nuovo.

SUPER LIGHT PEN

Eccezionale penna ottica di alta qualità, sfrutta componenti elettroluminosi eccellenti per una precisione millimetrica che non si riscontra assolutamente nelle concorrenti; disegnare col computer diventa più facile e veloce, mantenendo la precisione che può dare un ottimo mouse. Il rapporto qualità-prezzo è favorevolissimo.

Per Amiga o per PC con scheda grafica 99.000

ROBOT-ARM

Esplora anche tu le meraviglie della robotica, con questo completo braccio automatico; dispone di ben 5 assi di movimento, per una versatilità unica! Controllato semplicemente da una coppia di joystick, oppure dal computer con l'apposita interfaccia (opzionale), che permette di creare lunghe serie di movimenti da far ripetere al braccio oltre ad altre grandi possibilità di programmazione. Completo di accessori come pinze, pale, stabilizzatori ed attacchi magnetici.

Robot-Arm 159.000
Interfaccia per Amiga 89.000
Interfaccia per C64 .... 89.000

PORTADISCHETTI

Nuovi modelli in ABS antiurto, antistatico, antiacido ed autoestinguente, dal design modernissimo, geniale e decisamente bello. I modelli più piccoli (portatili) hanno una apertura a ventaglio per consentire una ricerca veloce dei dischetti, ed una chiusura completamente ermetica.

Modello	Prezzo
3"1/2 5 pz. colorati	4.000
3"1/2 10 pz.	5.000
3"1/2 20 pz.	15.000
3"1/2 40 pz.	20.000
3"1/2 80 pz.	28.000
3"1/2 Posso 150 pz.	39.000
5"1/4 8 pz. colorati	5.000
5"1/4 50 pz.	22.000
5"1/4 100 pz.	26.000
5"1/4 180 pz. Posso	44.000

COPERTINE PER COMPUTER

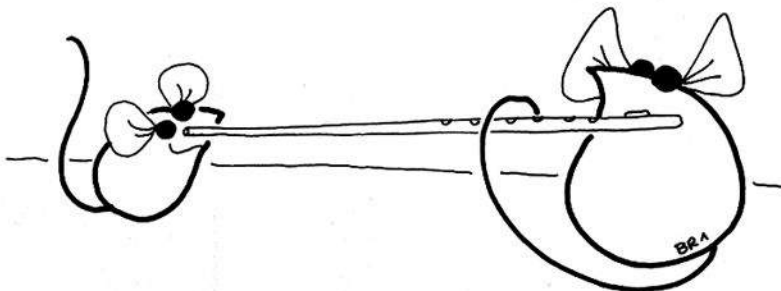
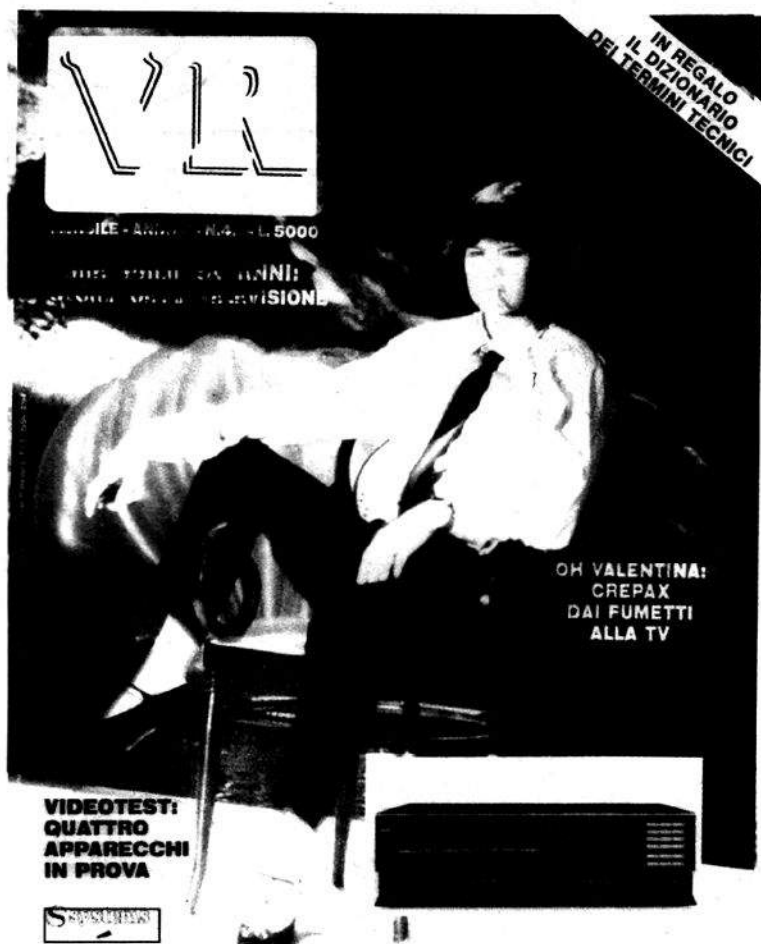
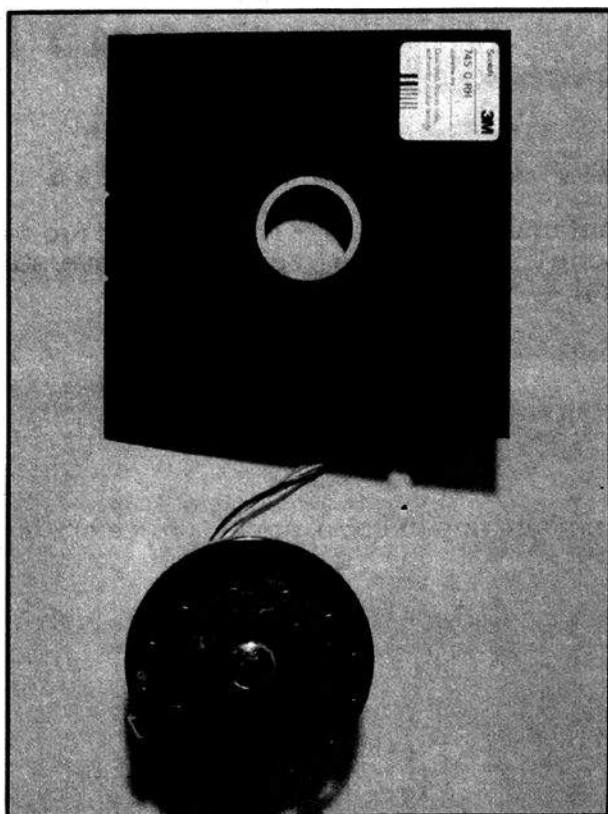
Indispensabili per proteggere il vostro computer,monitor e stampante da polvere, graffi, liquidi o umidità che potrebbero danneggiare i vostri investimenti ed il vostro lavoro. Realizzati su misura per decine di marche diverse, sono disponibili nelle versioni in PVC di colore grigio metallizzato, lavabile, antistatico, autoestinguente e molto robusto. In alternativa, per le sole tastiere, sono disponibili in plexiglass rigido trasparente, oppure ancora una pellicola trasparente che va stesa sopra i tasti, elegante ed innovativa, poichè permette di lavorare continuando a proteggere la tastiera.

Diversi prezzi per i vari modelli - telefonare

Viale Monte Nero 31
20135 Milano

Tel. (02) 55.18.04.84
(4 linee ric. aut.)
Fax (02) 55.18.81.05 (24 ore)

Negozio aperto al pubblico tutti i giorni dalle 10 alle 13 e dalle 15 alle 19.
Vendita per corrispondenza.
Sconti per quantità ai sigg. Rivenditori.

I prezzi Flopperia sono IVA compresa, sempre!

```
; * INSTALL MESSAGE   1989 by Massimiliano Alessandri

IF NOT EXISTS RAM:Copy
Echo "*NAttendi...sto caricando i comandi AmigaDOS nella RAM."
Copy c to RAM: All Quiet
Path RAM:
EndIF
IF NOT EXISTS SYS:s/Message
Echo "*NDevi creare un messaggio di default sul disco."
Echo "(entrata modo inserimento automatica)"
Wait 3
ED SYS:s/Message
EndIF
Copy SYS:s/Message RAM:
Ask "*NVuoi modificare o vedere il messaggio in RAM (y/n) ?"
IF WARN
ED RAM:Message
EndIF
Echo "*NPer favore inserisci il disco dove vuoi il messaggio."
Ask "Poi premi il tasto RETURN."
CD DFØ:
IF NOT EXISTS s
Makedir s
EndIF
IF NOT EXISTS c
Makedir c
EndIF
Copy RAM:Type c
Copy RAM:Message s
Ask "*NVuoi installare il disco (y/n) ?"
IF WARN
Install dfØ:
EndIF
Echo "*NOra inserisci manualmente nel file s/Startup-Sequence"
Echo "di questo disco, nel punto ove desideri, il comando"
Echo "TYPE S/MESSAGE. Entrata modo inserimento automatica."
Wait 5
ED s/Startup-Sequence
FileNote s/Message "Erase this file and you will corrupt main program data !"
Ask "*NVuoi proteggere tutti i files dalla cancellazione (y/n) ?"
IF WARN
Protect s/Message
Protect c/Type
Protect s/Startup-Sequence
EndIF
CD SYS:
```

PRONTO, CHI BYTE?

Vediamo di capire un po' che cosa vuol dire telematica

di **Ascanio Orlandini**

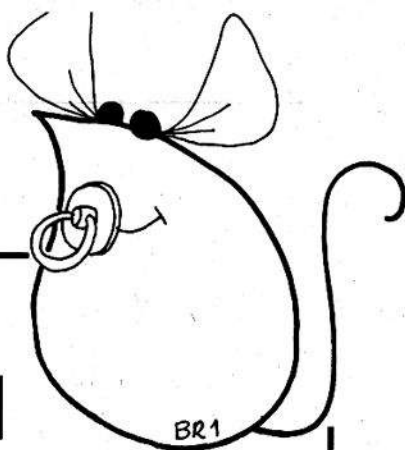
Vi sono molti possessori di computer che non hanno la benchè minima idea del significato celato sotto la parola "telematica". Lo scopo proposto è proprio quello di chiarire il significato del termine e, soprattutto, quello di invitare nuovi membri nell'universo telematico in modo da diffondere ciò che non è una moda, ma bensì un modo giovane ed intelligente per usare i computer per divertirsi, ma anche per lavorare.

Ma che cos'è la telematica? E' quel... qualcosa che permette di inviare dati a distanza con l'ausilio delle normali linee telefoniche. La possibilità di inviare dati, documenti e informazioni a distanze anche notevoli (pensate alla teleselezione intercontinentale) è ormai uno strumento per agevolare il lavoro di ufficio e velocizzarlo.

La telematica "pratica" è nata, si può dire, con l'invenzione del Telex e si è poi sviluppata grazie ai Modem, sempre più funzionali e sofisticati. L'ultimo parto di questa scienza in continuo progresso è il Telefax che permette di inviare e ricevere documenti, grafici e immagini. Tra tutti gli strumenti sopracitati, il più flessibile è senz'altro il modem che non ha limiti di testi o immagini come i suoi fratellastri, ma che, essendo complementare ai computer, permette di trasferire di tutto: da semplici testi a complesse immagini elaborate ed addirittura programmi di ogni tipo per ogni computer esistente.

Un utilizzo serio è quindi facilmente intuibile. Oltre a collegare attivamente sedi ed uffici tra loro, c'è la possibilità di collegarsi a Banche Dati che permettono di disporre di servizi altamente professionali, di solito a disposizione solamente delle grandi aziende. Tramite la Peis, ad esempio, si possono ottenere nell'arco di 24 ore (e dietro modico compenso) dati e bilanci relativi ad altre aziende, verifiche protesti, trasmissione / ricezione di fax e telex, informazioni su nuove disposizioni legislative, consigli legali e tanto altro ancora.

Molto sviluppata è comunque anche la telematica amatoriale e hobbistica. A que-

PRINCIPIANTI, I DIECI COMANDAMENTI

1 Leggi attentamente il libretto di istruzioni del tuo computer e delle periferiche che ti sei procurato (registratore, drive, stampante, monitor, eccetera)

2 Non limitarti a leggere i listati di esempio ivi riportati, anche se sembrano banali: digitali sul computer e falli girare (digitando Run e premendo il tasto Return). Se non ti sono chiari, al contrario, la loro digitazione (e successiva esecuzione) ti chiarirà le idee.

3 Nel digitare i listati, ricordati di premere SEMPRE il tasto Return quando giungi alla fine del rigo, anche se tale operazione può sembrarti inutile.

4 Non confondere la vocale alfabetica "O" con il numero zero "0".

5 Digita sempre per esteso il comando PRINT e non abbreviarlo mai con il punto di domanda (?).

6 Inizia a digitare i listati più semplici e brevi: quelli più lunghi potrai digitarli quando avrai acquisito una maggior dimestichezza con il computer.

7 Dopo aver digitato un QUALSIASI programma, registralo subito, seguendo le istruzioni riportate sul manuale, PRIMA di dare Run.

8 Dopo aver fatto partire un programma, in caso, ad esempio, di segnalazione di errore in linea 350, digita soltanto...
List 350
...e accertati che la linea che appare sul video sia RIGOROSAMENTE identica a quella stampata sulla rivista.

9 Se, digitando un listato, ti accorgi che vi sono istruzioni di tipo Poke e Sys, raddoppia la prudenza nella digitazione e nelle procedure di registrazione PRIMA di dare Run.

10 Accertati che il programma che ti accingi a digitare sia REALMENTE valido per il tuo computer. Su ogni articolo della nostra rivista (da leggere SEMPRE con attenzione) è indicato il tipo di computer per il quale il listato stesso è idoneo.

sto scopo sono state adibite centinaia di banche dati pubbliche in tutta Italia, di libero accesso e totalmente gratuite. Queste banche dati, o meglio, questi bbs (Boulettin Board Systems), constano generalmente di due parti ben distinte e separate.

La prima è riservata alla messaggistica e la seconda allo scambio di files e programmi (rigorosamente di pubblico dominio). Da poco più di due anni si è ottimizzata una vera rete italiana di "FIDO BBS" di ben 86 nodi (peraltro in continua crescita) sparsi in tutto la nazione e continuamente interconnessi tra loro.

Ciò vuol dire che un messaggio che lasciate in una bbs installata nella vostra città (grazie, magari, ad un solo scatto SIP) può fare il giro di mezza Italia e voi non dovrete fare altro che aspettare le risposte che arriveranno sul "vostro" bbs. Tale comodità diventa particolarmente utile se dovete vendere / acquistare qualcosa (nell'apposita area "mercatino") o chiedere aiuto per quanto riguarda particolari problemi per il vostro sistema (nelle aree appositamente disposte per tutti i computer più diffusi, inclusi tutti i "fratelli" Commodore).

Ma veniamo ora alla funzione che dall'esterno probabilmente è considerata la più importante: la possibilità di trasferire files e programmi con la sola spesa della chiamata telefonica; processo questo relativamente lento, se attuato con un modem da 300 baud, ma che si velocizza alquanto passando ai 1200, diventando una scheggia a 2400.

Per chi proprio non sapesse il significato di "baud" chiariremo semplicemente il termine definendolo come la velocità con cui avviene il collegamento (bit / secondo). Per quanto riguarda la disponibilità di software, beh, di certo ce n'è a sufficienza, calcolando l'elevato numero di bbs, e per tutti i gusti e sistemi (dal glorioso C/64 al C/128, dall'MS-DOS all'Amiga) e ognuno può trasmettere propri programmi, files, disegni, e così via a disposizione degli altri utenti compiendo l'operazione inversa al prelevamento (in gergo: upload).

Esistono anche bbs dedicate interamente all'Amiga, o specializzate in musica piene zeppe di suoni digitalizzati, di composizioni musicali, di effetti sonori utilizzabili con i più diffusi packages.

L'ingresso nella telematica risulta quindi essere un passo obbligato per chi ritiene che la propria macchina non sia solo un videogame e vuole svilupparla attivamente in modo non individualistico, ma bensì insieme a tantissimi amici che coltivano l'interesse comune, amicizia che nasce attraverso una tastiera ed un monitor e che non di rado si trasforma in amicizia "vera".

E poi, dite la verità, chi non ha mai provato un certo fascino nei confronti della tecnologia telematica assistendo a films come War Games? Per tuffarsi in questo universo infinito non occorre nulla di particolare: un computer semplicissimo (addirittura il mitico ed ormai introvabile VIC-20 potrebbe bastare allo scopo!) corredato di un semplice modem con relativo programma di comunicazione ed il gioco è fatto!

Per chi possiede un Commodore 64, bata procurarsi l'Adattatore Telematico 6499 che "Mamma Commodore" ha messo sul mercato. Non è il massimo della vita, d'accordo, ma è più che sufficiente da quando si è riusciti a superare il grande handicap dell'impossibilità di trasferire programmi tramite un piccolo package (di pubblico dominio), presente nella lista dei programmi prelevabili di molte bbs, che implementa il protocollo di trasferimento X-modem, sicuramente il più diffuso; modem che è inoltre predisposto per collegarsi al VIDEOTEL, il servizio telematico offerto dalla SIP, funzionante a 75 / 1200 baud (riceve a 75 ed invia a 1200), molto ben fornito, organizzato ed assistito, che per ha il difetto di essere... a pagamento.

Chi è, invece, fortunato possessore di Amiga o Pc Ms-Dos compatibile, può accedere ad una vastissima gamma di prodotti, che le varie Aziende immettono sul mercato, dai prezzi molto variabili e prestazioni direttamente proporzionali al prezzo.

Ricordatevi per che se per un C/64 possono bastare i 300 baud, per gli altri computer tale velocità non è sufficiente a causa della maggior lunghezza dei programmi intercambiabili. Sempre a meno che siate disposti a passare la notte attaccati ad una bbs, tenendo presente che il 99.9(9)%, per non dire il 100%, ha un tempo d'accesso limitato dai 30 ai 60 minuti.

Speriamo, con questo breve articolo, di avere invogliato qualcuno ad entrare nell'universo telematico che, superato il breve shoc dell'acquisto dell'attrezzatura adeguata, è molto appagante, parola di telematico! Per ogni eventuale commento, suggerimento o consiglio, potete contattare l'autore del presente articolo su Euro Elettronica Bbs.

Nodo nr.	Nome	Regione		Numero telefono	Velocita massima
Region 33	Italy 1	Potenza	PT	0971.35447	1200
1	Fido Pz	Potenza	PT	0971.35447	1200
2	Fido PEG	Potenza	PT	0971.35447	1200
100	Quick Bbs Support	Torino	TO	011.5765565	1200
101	Lynx Support Italy	Rimini	RI	0541.27135	2400
201	Ita Echo Coord	Milano	MI	02.2666502	2400
Host 331	NorthernC Ita.	Milano	MI	02.2666502	2400
1	Ipotesi	Milano	MI	02.2666502	2400
2	Fido-Mi	Milano	MI	02.33000153	2400
Down 3	D.M.B.	Milano	MI	02.316824	9600
4	Prisma	Cremona	CR	0372.436900	1200
6	N-E-M-O	Rozzano	MI	02.8245137	1200
7	Blue Net	Novara	NO	0321.28929	1200
8	AtekLink	Brescia	BS	030.9719440	2400
9	Opus Varese	Varese	VA	0331.263425	2400
10	**Euro Elettronica**	**Crema**	**CR**	**0373.86966**	**2400**
11	Amiga Line	Brescia	BS	030.2420452	2400
12	BBS 2000	Milano	MI	02.706857	1200
13	OPUSNOVA	Sondrio	SO	0342.493782	1200
14	Howard The Duck's	Milano	MI	02.6551412	1200
15	Clessidra-New	Milano	MI	02.4159728	1200
16	Lario Net	Mandello	CO	0341.735693	1200
17	TeleSiBoc	Milano	MI	- Privata -	2400
18	HAM Link	Milano	MI	02.33402660	1200
19	CittÇ Informa	Lecco	CO	0341.361064	1200
20	Fiscal Data Bank	Milano	MI	02.6697700	1200
21	N.E.C. Fiscal News	Milano	MI	02.6697754	1200
22	Opus Bergamo	Bergamo	BG	035.904032	1200
23	Servidati BBS	Novara	NO	0321.450894	2400
24	HAL-bbs	Varese	VA	0332.286849	1200
Host 332	Italy 88-Telenet			055.253606	2400
1	Bit-Show	Parma	PR	0521.38982	1200
Down2	Bit-Show-2	Parma	PR	0521.285014	1200
3	SC-Link Scandicci	Firenze	FI	055.253606	2400
4	Digic-Link	Firenze	FI	055.282365	2400
5	Opus-Rapallo	Rapallo		0185.274020	1200
6	Genova 2000	Genova	GE	010.3770080	2400
7	FIDO Rimini	Rimini		0541.773527	2400
Down8	FIDO Livorno	Livorno	LI	0586.884380	1200
9	OCA System	Bologna	BO	051.6343719	2400
10	Blue Sea Bbs	Genova	GE	010.3770365	2400
11	Datatel Carpi			059.688994	2400
12	Utopia	Pistoia		0573.368164	2400
13	OPUS BELUSHI	Albenga		0182.52329	2400
14	CPUlink	Prato		0574.433345	2400

15	Blues BrotherS	Genova	GE	- Privata -	1200
16	ARCI Computer Club	Bologna	BO	051.515311	2400
18	Opus LI	Livorno	LI	0586.501074	1200
19	Opus Guastalla	Guastalla		0522.824379	1200
Host 333	Fri-Ve-Net	Pordenone	PN	0434.32020	2400
1	Fido PN	Pordenone	PN	0434.32020	2400
2	Fido Padova 1	Padova	PD	049.663-452	2400
3	Fido Padova 2	Padova	PD	049.620-035	2400
5	The Wall	Vicenza	VI	0444.961708	1200
6	Bit One	Verona	VE	045.686-0307	1200
7	The Clivius Link	Verona	VE	045.565988	1200
8	Opus Plat One	Bibione		0431.438-271	2400
9	Uni Opus	Verona	VE	045.7731283	1200
10	Bit One][	Verona	VE	- Privata -	2400
11	Akron Bbs	Pordenone	PN	0434.522555	1200
12	Asem Link	Buia		- Privata -	1200
13	Fox Bbs	Trento	TN	0461.821400	1200
Host 334	NorthWest-Italy	Torino	TO	011.5765-565	1200
1	Fido TO	Torino	TO	011.5765-565	1200
2	Opus Montecastello	Alessandria-AL		0131.355506	2400
3	Opus Tecnocity	Torino	TO	011.4115173	2400
4	Eporedia IVREA	Ivrea		0125.611624	2400
5	OpusLandia	Torino	TO	011.290312	2400
6	Torino-net#1	Torino	TO	011.539456	2400
7	Primula Rossa	Alessandria-AL		0131.42467	1200
8	Opus Poirino	Poirino	TO	011.9452705	2400
9	Torino-net#2	Torino	TO	011.538601	2400
10	PC-Opus	Torino	TO	011.3352858	2400
11	Charlie's Puppies	Torino	TO	011.364489	1200
800	Travelmatic	Torino	TO	011502423	2400
Host 335	CentroSud Italy	Potenza	PT	0971.35447	1200
1	Fido PZ	Potenza	PT	0971.35447	1200
	S. Maria CV Bbs	S. Maria		0823.812533	2400
3	Fido Roma	Roma	RM	06.9035120	1200
4	Opus The World Bbs	Napoli	NA	081.7433830	1200
5	Civitavecchia Bbs	Civitavecchia		0766.35352	1200
6	Alex Opus	Roma	RM	06.7482648	1200
7	Civitavecchia Bbs 2	Civitavecchia		0766.35352	1200
8	Smoke In The Hayes	Afragola		- Privata -	1200
9	DSH Soft Corp 1987	Roma	RM	06.8276490	2400
10	Line Bank	Cassano		0776.270573	1200
11	List Bbs Roma	Roma	RM	06.7665495	2400
12	Opus Mimac Bbs	Roma	RM	06.270400	1200
13	Logica	Civitavecchia		- Privata -	1200
200	Hub Napoli-S.Maria	S. Maria		0823.812533	2400
300	Hub Roma	Roma	RM	06.9035120	1200
400	Hub Civitavecchia	Civitavecchia		0766.22077	2400

SEQUENZIALI A TUTTO GAS

Sfruttando a fondo le caratteristiche dei drive dedicati al C/128, è possibile implementare prestazioni impensabili; come, per esempio, un fastload molto, molto speciale...

di **Domenico Pavone**

Come i veterani di fede Commodoriana ben sapranno, la velocità di accesso ai dati memorizzati su periferica ha subito un lento, ma costante progresso.

Per la categoria degli home ad otto bit, si è infatti passati dall'arcaico tape ad "effetto notte" (visti i tempi di attesa), ai ben più confortevoli blocchi del drive 1541.

Anche in questo caso, però, dopo un minimo di assuefazione, e visto il proliferare di megaprogrammi a tutto disco, la compatta truppa di sessantaquattristi (giustamente mai sazia) ha ricominciato a dar fiato alle trombe: "Ma che drive è, se bisogna aspettare tanto per caricare un centinaio di blocchi?".

Ed ecco fiorire la miriade di protesi (stiamo parlando di cartridge, non di dentiere) Turbo, Speed, o Fast che più Fast non si può, diventate quasi un tuttuno con lo chassis del C/64.

Con l'avvento del C/128, anche se non per tutti, le cose sono cominciate a cambiare davvero.

Finalmente, tra le Rom dell'ex gioiello-rivelazione (sappiamo tutti com'è andata a finire) di casa Commodore, viene inserito tutto il necessario per un accesso veloce ai drive, purchè questi siano in grado di interagire con le routine di sistema.

In altre parole, purchè il 128 sia collegato ad un drive modello 1570 o 1571.

Tuttavia, anche per chi appartiene a questa (quasi) fortunata categoria di utenti, quello che comunemente viene definito FastLoad ha un preciso limite: è attivo solo in rapporto a comandi come Dload o Bload.

In pratica è applicabile unicamente a file di tipo programma (Prg).

Molto spesso, d'altra parte, ci si trova di fronte alla esigenza di trattare dei dati memorizzati in formato sequenziale (Database, Word Processor, ecc.), con l'obbligo di ricorrere a comandi decisamente più "pigri": Get# ed Input#, associati ai ben noti Open e Close.

Tra l'altro, se adoperato in maniera ortodossa, anche il linguaggio macchina non può essere molto di aiuto: il trattamento dei dati diverrà ultraveloce, ma l'accesso al drive rimarrà quello che è.

Un esempio in tal senso può essere rilevato nella routine di lettura di files sequenziali pubblicata sul n. 56 della rivista (pagina X di Campus), la cui velocità non è influenzata dal modello di drive posseduto.

Un modo di aggirare l'ostacolo, però, esiste, ed è legato ad una delle caratteristiche meno chiare del manuale in dotazione ai drive 1570/71: i cosiddetti Burst Commands.

La maggior parte di essi, data la carente documentazione, risulta alquanto dura da digerire, e comunque può trovare applicazione solo in casi estremamente improbabili per un programmatore di livello medio-alto.

Tuttavia, a beneficio di chi volesse cimentarsi nell'arduo compito, in questa sede verrà chiarito il funzionamento di uno di questi speciali comandi, aprendo il campo ad eventuali sperimentazioni sugli altri.

Ma, come sempre, prima di tutto la pratica.

TURBOSEQ

Si copi, dunque, il listato di queste pagine, e lo si salvi su disco prima di mandarlo in esecuzione con Run.

PROBLEMI DI BANCO

La routine proposta in queste pagine memorizza il contenuto di un file in banco 0.

La scelta non è occasionale, in quanto consente di effettuare switch indolori tra il banco contenente le routine di sistema, cioè il 15, ed il banco 0, normalmente adibito a contenere il testo dei programmi basic.

Per passare senza problemi da una configurazione all'altra, è infatti obbligatorio che una routine sia presente nelle stesse locazioni di entrambe.

Il codice macchina di Sload, allocato nelle locazioni da $1300 in poi, risponde pienamente a tale condizione, per cui si rende possibile una procedura come quella implementata alle righe 149-152 del disassemblato.

In pratica viene prima impostato bank 0, poi viene "storato" il contenuto dell'accumulatore, ed infine ripristinato bank 15.

Per commutare di banco in linguaggio macchina, come dovrebbe essere noto, basta inserire l'appropriato valore nella locazione $FF00 (0 = banco 15, $3F = banco 0, $7F = banco 1), come p.es. viene fatto nelle righe 151 e 152.

Ma esiste un altro modo, molto più immediato.

Se con il Monitor si esaminano le locazioni immediatamente successive ad $FF00 (di qualsiasi banco), si noterà che sono presenti proprio i valori $3F, $7F e $01.

Ebbene, basta accedere in scrittura, con un valore qualsiasi, ad una di queste locazioni, per rendere attivo il banco desiderato.

In pratica, come avviene in riga 149, con Sta $FF01, quale che sia il contenuto dell'accumulatore, si passerà in banco 0: esattamente come se il contenuto di $FF01 ($3F) fosse stato depositato in $FF00.

Analogamente, se avessimo pokato qualsiasi cosa in $FF02 (contenuto = $7F), ci si sarebbe trovati immediatamente in bank 1.

Nel nostro caso, però, con esiti disastrosi, visto che da lì il sistema non sarebbe potuto tornare alla routine.

Comodo, vero?

Come testimoniato dalla discreta mole di Data, siamo in presenza di un programma "caricatore", che si limita ad installare in memoria una routine in linguaggio macchina a partire dalla locazione $1300 (decimale 4864), nonchè ad attivarla mediante la Sys di riga 160.

Se tutto è in regola, da questo momento è reso disponibile un nuovo comando basic, Sload, da usarsi secondo la sintassi...

Sload "nomefile"

...con nomefile che obbligatoriamente deve far riferimento al un file di tipo sequenziale (SEQ) presente sul dischetto in opera.

Qualora fosse presente un file col nome indicato, ma di tipo Prg, verrebbe solo generata una (consueta) segnalazione di errore da parte del drive.

Compito di questo comando, molto simile a Bload, è quello di caricare direttamente in memoria il contenuto di un file sequenziale, esattamente come se si trattasse di un programma, ed alla stessa velocità.

Il che, trovandoci al cospetto di drive ultraveloci, significa poco più di un paio di secondi per files di un centinaio di blocchi (vi sembra poco?).

Se a questo si aggiunge la possibilità di utilizzare il comando all'interno dei propri programmi basic, ecco che la creazione di archivi personalizzati, con accesso ultraveloce ai dati, diventa una realtà aperta a tutti.

Per conoscere più a fondo Sload, e verificarne i vantaggi, ricorriamo ad un esempio.

Dopo avere installato in memoria la routine lm, si digiti questo breve listato...

```
10 A$ = "dato": open 5, 8, 5, "demo, s, w"
20 for x=1 to 500: print#5, a$ + str$(x)
30 next x: close 5
```

...e lo si mandi in esecuzione.

Verrà così creato, sul disco presente nel drive, un file sequenziale (di nome Demo) contenente 500 stringhe "dato X", con X che assumerà tutti ivalori da 1 a 500.

Volendo evitare l'attesa necessaria per creare questo file, può eventualmente essere utilizzato qualunque altro file di tipo seq in vostro possesso, purchè sia noto quanto vi è memorizzato.

Si provi ora a leggere il contenuto del file Demo, adoperando un metodo tradizionale: si impartisca New, quindi si digiti ancora...

```
10 Open 5, 8, 5, "demo"
20 get#5, a$
30 if st <> 64 then 20
40 close 5
```

Dopo il Run, il file verrà letto "a vuoto", nel senso che il suo contenuto non viene conservato, per rendere il più veloce possibile la procedura (di norma, occorre ancora più tempo, per esempio, per trasformare la stringa in numero, ed eventualmente pokarlo da qualche parte).

Ora, si provi ad impartire:

Sload "demo"

La differenza risulterà palpabile, e lo sarebbe ancora di più in presenza di file molto più corposi (provate per credere).

Ma... dov'è finito il contenuto del file? Vediamolo subito.

Si entri in ambiente monitor (Shift + F7), e si impartisca M 08000.

In reverse, ecco lì i cartteri ascii delle prime stringhe memorizzate nel file demo.

Quindi, come avrete già capito, un file caricato con Sload viene depositato in memoria a partire dalla locazione $8000 del banco 0, lo stesso che contiene il testo dei programmi Basic.

Da lì può essere poi manipolato come più ci aggrada: tramite Peek, o ancora meglio con opportune routine in linguaggio macchina.

La locazione di inizio del caricamento, può comunque essere modificata con...

Poke 5027, low: Poke 5029, hi

...con low ed hi che indicano l'indirizzo voluto (sempre di banco 0), suddiviso nel suo byte basso e byte alto.

Si ricorda, a tal proposito, che questi due valori possono essere ricavati dalla formula:

HI = INT (indirizzo / 256)
LOW = indirizzo -HI * 256

OCCHIO ALL'ERRORE

Nonostante la sua (mai sufficiente) completezza, il nuovo comando aggiunto al basic 7.0 va adoperato con una certa accuratezza.

Come si potrà meglio constatare nella descrizione del disassemblato della routi-

ne che lo implementa, Sload è in grado di riconoscere eventuali errori che lo coinvolgono, come "File Not Found" o "String Too Long".

Inoltre è stato aggiunto un controllo sull'eventuale straripamento (oggi va più di moda tracimazione) del file nelle locazioni di memoria che vanno da $F000 in poi.

Qualora si verificasse, si avrebbe un ritorno al basic con una (voluta) segnalazione "Out Of Memory", mentre la spia del drive potrebbe anche restare accesa in permanenza: in questo caso, basterà impartire in modo diretto Dclear per resettare il drive, e tutto tornerà normale.

E' buona norma, comunque, prevedere sempre l'occupazione che avrà in memoria il file caricato, soprattutto se si intende usare Sload all'interno di propri programmi.

Un'altra segnalazione di errore, stavolta non convenzionale, è rappresentata da "Drive Non Fast", che apparirà se il C/128 è collegato ad una periferica tipo 1541, non abilitata alla comunicazione veloce, o anche se si è impartito (ad un drive 1570/71) il comando "U0)M0".

Quest'ultimo, infatti, anche se solo via software, renderebbe un 1570/71 del tutto simile ad un 1541, e quindi inadatto al FastLoad.

Per concludere, non resta che segnalare l'unico (si spera) tallone d'Achille del comando: usando al posto del nome del file degli spazi vuoti (bisogna proprio andarsela a cercare!) racchiusi tra virgolette, non resta che... resettare il computer.

In questo remoto caso, basterà impartire nuovamente Sys 4864 per riavere attivo il comando Sload.

PRONTA CONSEGNA

La ricezione di dati attraverso il bus seriale comporta in teoria una complessa gestione delle temporizzazioni interne del sistema.

In pratica, però, la maggior parte del lavoro viene svolta (piuttosto egregiamente) dal computer.

Quando, come nel caso del comando Sload, è necessario interagire con la trasmissione di singoli byte da periferica (drive) Fast ad unità centrale, gli unici elementi da prendere in considerazione sono 3 registri del CIA.

Prima di tutto $DD00, o meglio il suo bit 4.

Questo, al momento della ricezione, deve risultare settato (righe 165-167 del disassemblato).

La trasmissione di un byte sul bus seriale, avviene bit per bit.

Ebbene, quando l'intero byte è stato trasmesso, viene generato un segnale di interrupt, che può essere rilevato testando il bit 4 di un altro registro, $DC0D.

Se questo risulta settato, in pratica significa che il byte trasmesso dal drive è pronto... per la consegna, presente nella locazione $DC0C.

Il registro degli interrupt del CIA ($DC0D), presenta inoltre la caratteristica che, se letto (p.es. con Lda), il suo contenuto viene azzerato.

Nella nostra routine, il tutto viene applicato come segue:

Alle righe 107-111 è azzerato preventivamente il bit 4 del registro di I/O, nonchè il registro degli Interrupt del CIA (quest'ultimo con la semplice lettura Lda $dc0d).

Poi, al momento della ricezione vera e propria del byte trasmesso (righe 165-172), lo stato del bit 4 di $DD00 viene invertito (e quindi posto ad 1), e si continua a testare il bit 4 di $DC0D, finchè non ci segnalerà che tutto il byte è stato trasmesso.

A questo punto, un semplice Lda $DC0C renderà disponibile il dato in accumulatore, pronto per essere trasferito in memoria dalle istruzioni alle righe 149-152.

UN BURST DA SCOPRIRE

Vediamo ora di capire come funziona il tutto, esaminando il disassemblato della routine, espresso in un formato simbolico elaborato con il MacroAssembler Commodore (già, proprio quello del C/64).

Chi volesse vederlo con i suoi indirizzi effettivi, non avrà che da installare la routine in memoria, ed esaminarla con il comando D del Monitor in dotazione al C/128.

La prima parte del listato Assembly (righe 15 - 59) si occupa di gestire il nuovo comando, usando la tecnica di dirottamento della routine di errore ampiamente descritta nel già citato articolo apparso sul n. 56.

Pertanto, chi volesse approfondirla, non ha che da ripescare il non troppo lontano numero di CCC, o affidarsi ai commenti presenti nel disassemblato.

A seguire, inizia la messa in opera della Fastload Utility, descritta (si fa per dire) a pagina 87 del manuale inglese del drive 1570/71.

In pratica, si tratta di inviare al drive, tramite il consueto canale di comando (per intenderci: dopo un open 15, 8, 15) una serie di tre byte.

I primi due, sul manuale descritti in notazione binaria, altro non sono che la traduzione Ascii dei caratteri "U0".

Il terzo byte, invece, deve essere così composto:

X0011111

Il bit più significativo (X), indica il tipo di file da caricare.

Se azzerato, va riferito a files di tipo PRG; settato (=1), accetterà il caricamento di file SEQ.

Il primo caso, com'è ovvio, non ci interessa più di tanto: esistono già Bload e Dload.

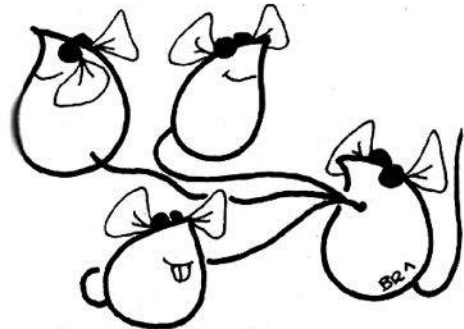
Inviando invece un byte di valore 159 (1001.1111 in binario), apriamo la strada al nostro nuovo comando.

Dopo questo byte, è necessario inviare, uno ad uno e sempre sul canale di comando del drive, i caratteri che compongono il nome del file.

Tutte queste fasi, sono chiaramente individuabili nelle righe da 61 a 99, che comprendono anche un controllo sul tipo di drive collegato (85-87).

A tale scopo, viene testata la locazione $A1C, dopo l'invio al drive di un segnale Untalk (routine del Kernal $FFCC); se la periferica è di tipo Fast, il bit 6 della suddetta locazione risulterà settato, in caso contrario la nostra routine stampa la stringa "Drive non fast" (89-94) e ritorna al basic.

128 CHIAMA DRIVE

A questo punto, inizia la fase (apparentemente) più complessa: il prelevamento

dal file del suo contenuto, leggendolo un byte alla volta.

La procedura di ricezione, deve però tenere presente le modalità con cui il drive invia il "materiale".

Il primo byte letto, nonchè il primo di ogni settore da leggere, rappresenta il byte di stato.

Se questo è uguale a 2, significa che il file non è stato trovato (oppure era di tipo Prg).

Se, invece, contiene il valore 31 (EOI), indica che siamo di fronte all'ultimo blocco del file; in questo caso, il successivo byte inviato specificherà quanti byte del settore sono occupati dal file.

Valori superiori a 2, e diversi da 31, indicano tutta una serie di errori descritti sul manuale.

In pratica, quello che la nostra routine deve fare (righe 113-133), può essere così riassunto:

1) Preleva il byte di stato e lo controlla.
2) Se Status è minore di 2 e diverso da 31, legge i 254 byte che costituiscono un settore (i 2 di link sono trattati automaticamente dal sistema).
3) Se Status = 31, legge il prossimo byte (numero di byte del prossimo settore), e quindi passa a ricevere l'ultima porzione di settore occupata dal file prima di tornare al basic.
4) Se Status = 2, esce con File Not Found.
5) Se Status è maggiore di 2 e diverso da 31, esce con errore generico.
6) Torna al punto 1.

Nel disassemblato, oltre a queste fasi, viene anche attivato il modo Fast (riga 110), e preparati (con l'indirizzo $8000, eventualmente modificabile) due puntatori in pagina zero (102-105) per pokare i byte ricevuti nelle volute locazioni di memoria (149-163) di bank 0 (vedi riquadro).

La lettura di un blocco, e relativo storaggio in memoria, viene eseguita dalla subroutine presente alle righe 146-163, e può essere facilmente compresa seguendo i commenti del disassemblato.

Al suo interno, viene inoltre chiamata un'altra subroutine, LegByt (165-172), che si occupa di prelevare un singolo byte dal file (vedi riquadro).

Chi ha già una certa dimestichezza con l'Assembly, potrebbe migliorare la routine aggiungendo una rilevazione di tutti gli errori segnalati dal byte di stato, ma, anche così com'è, il nostro spartano Sload può risultare utile nelle più svariate applicazioni.

Magari, fateci sapere... com'è andata.

```
    DISASSEMBLATO IN FORMATO SIMBOLICO
------------------------------------------
001 ERRVTT  =$300          dec.768
002 BASPN   =$3D           dec.61
003 ERRSYS  =$4D3F         dec.19775
004 FAST    =$E5C3         dec.58819
005 OPEN    =$FFC0         dec.65472
006 SETFIL  =$FFBA         dec.65466
007 SETNAM  =$FFBD         dec.65469
008 OUTFIL  =$FFC9         dec.65481
009 BSOUT   =$FFD2         dec.65490
010 CLRCHN  =$FFCC         dec.65484
011 CLOSE   =$FFC3         dec.65475
012 BANK15  =$A845         dec.43077
013 LEN     =$FA           dec.250
014         *=$1300        Start a 4864
015 ;=====================================
016         LDA #<ROUT     Dirotta vettore
017         LDX #>ROUT     routine di er-
018         STA ERRVTT     rore a riga
019         STX ERRVTT+1   25 (rout) e
020         RTS            torna al basic.
021 ;=====================================
022 ERR23   LDX #23        String too long.
023 ERR     JMP ERRSYS     Esce ad Errore.
024 ;-------------------------------------
025 ROUT    CPX #11        Sintax Error?
026         BNE ERR        Se no, Errore.
027         CMP #147       Token di Load?
028         BNE ERR        Se no, Errore.
029         DEC BASPN      Basic=Basic-1.
030 ;-------------------------------------
031         LDA BASPN      Controlla se
032         CMP #$FF       cambio di pagi-
033         BNE CONFR3     na, ed aggiorna
034         DEC BASPN+1    puntatore alto.
035 ;-------------------------------------
036 CONFR3  JSR $3C9       Testo corrente
037         CMP #'S        uguale ad "S"?
038         BNE ERR        Se no, Errore.
039         JSR $380       Avanza di due
040         JSR $380       byte nel testo.
041         CMP #34        Virgolette?
042         BNE ERR        Se no, Errore.
043 ;=====================================
044         JSR $380       Prossimo byte.
045         LDX #0         Legge carattere
046 LOOP1   JSR $3C9       corrente.
047         CMP #34        Virgolette?
048         BEQ CONT1      Se si', salta.
049         STA NOME,X     Deposita byte.
050         INX            Controlla se
051         CPX #17        nomefile > 16.
052         BCS ERR23      Se si', Errore.
053         INC BASPN      Incrementa pun-
054         BNE LOOP1      tatori al testo
055         INC BASPN+1    e continua let-
056         BNE LOOP1      tura nomefile.
057 ;-------------------------------------
058 CONT1   STX LEN        Salva Len(nome).
059         JSR BANK15     Passa in bank15.
060 ;=====================================
061         LDA #15
062         LDX #8
063         TAY
064         JSR SETFIL     Open15,8,15.
065         LDA #0
066         JSR SETNAM
067         JSR OPEN
068 ;-------------------------------------
069         LDA $A1C       Azzera bit 6
070         AND #$BF       indicatore di
071         STA $A1C       Fast Serial.
072         LDX #15        File 15 aperto
073         JSR OUTFIL     in output.
074 ;-------------------------------------
075         LDA LEN
076         ADC #3         Legge ed invia
077         TAY            sul canale di
078         LDX #0         comando (15)
079 LOOP2   LDA COMAND,X   "U0", 159, ed
080         JSR BSOUT      i caratteri
081         INX            del nome del
```

```
082          DEY              file seq.
083          BNE LOOP2
084  ;-----------------------------------
085          JSR CLRCHN       Segnale al drive.
086          BIT $A1C         Se bit 6 di Fast
087          BVS BURST        Serial=1, salta.
088  ;-----------------------------------
089          LDX #0
090  LEGGE2  LDA MESG,X
091          JSR BSOUT        Legge caratteri
092          BEQ NOFAST       del messaggio
093          INX              fino allo zero.
094          BNE LEGGE2
095  ;-----------------------------------
096  NOFAST  LDA #15          Chiude canale
097          JSR CLOSE        di comando e
098          LDX #255         ritorna al basic
099          JMP ERRSYS       senza errore (x).
100  ;===================================
101  BURST   SEI              Disabilita IRQ.
102          LDA #$80         Inserisce in
103          LDY #00          due puntatori
104          STY $FB          ind. di inizio
105          STA $FC          del caricamento.
106  ;-----------------------------------
107          LDA $DD00        Azzera bit 4 del
108          AND #$EF         registro I/O
109          STA $DD00        del CIA 2.
110          JSR FAST         Modo Fast Input.
111          LDA $DC0D        Azzera reg. IRQ.
112  ;-----------------------------------
113  INIZIO  JSR LEGBYT       Legge stato.
114          CMP #2           Se <2 (tutto ok)
115          BCC MAIN         salta a routine.
116          BNE CONT2        Se <>2, salta.
117  ;-----------------------------------
118          LDA #15          Se =2 (file not
119          JSR CLOSE        found), chiude
120          LDX #4           canale 15, e
121          CLI              torna al basic
122          JMP ERRSYS       con errore 4.
123  ;-----------------------------------
124  CONT2   CMP #31          Se status <> 31,
125          BNE USCITA       esce, altrimenti
126          JSR LEGBYT       legge altro dato
127          TAX              (bytes ultimo set-
128          JSR ENTRY2       tore), e salta.
129  USCITA  CLI              Riabilita IRQ.
130          LDA #15          Chiude canale
131          JSR CLOSE        di comando.
132          JSR $380         Aggiorna testo, e
133          JMP $AF90        salta a Execute.
134  ;===================================
135  MAIN    LDA $FC          Raggiunto limite
136          CMP #$F0         della  memoria?
137          BCC VAI          Se no, Goto 143.
138          LDA #15          Se si', chiude
139          JSR CLOSE        canale comandi,
140          LDX #16          ed esce con er-
141          CLI              rore 16 (out of
142          JMP ERRSYS       memory).
143  VAI     JSR LEGBLK       Legge un settore
144          JMP INIZIO       e ricomincia.
145  ;====== S U B R O U T I N E S ======
146  LEGBLK  LDX #254         Bytes per blocco.
147  ENTRY2  LDY #0           Offset scrittura.
148  LEGGE   JSR LEGBYT       Legge un byte.
149          STA $FF01        Passa in bank 0.
150          STA ($FB),Y      Deposita byte.
151          LDA #0           Commuta memoria
152          STA $FF00        in banco 15.
153          INY              Continua a leg-
154          DEX              gere gli altri
155          BNE LEGGE        byte del blocco.
156  ;-----------------------------------
157          TYA              Aggiorna punta-
158          CLC              tori di pagina
159          ADC $FB          zero sommando
160          STA $FB          loro il numero
161          BCC RETURN       di byte preleva-
162          INC $FC          ti dal blocco.
163  RETURN  RTS              Return.
164  ;===================================
165  LEGBYT  LDA $DD00        Inverte bit 4
166          EOR #16          registro I/O
167          STA $DD00        del CIA.
168          LDA #8           Attende che un
169  ATTESA  BIT $DC0D        IRQ modifichi
170          BEQ ATTESA       bit 4 di CIA IRQ.
171          LDA $DC0C        Legge byte.
172          RTS              Return.
173  ;======= BUFFER  CARATTERI ========
174  MESG    .BYT 'DRIVE NON'
175          .BYT ' FAST',0
176  COMAND  .BYT 'U0',159
177  NOME    .BYT 0
178          .END
```

```
100 REM ----------------------------------------------------------------
110 REM       FASTLOAD DI FILES SEQUENZIALI PER C/128 + DRIVE 1570/1
120 REM ----------------------------------------------------------------
130 :
140 PRINT"ATTENDI":BANK15:FAST:FORX=0TO319:READA:B=B+A:POKE4864+X,A:NEXT
150 IFB<>38123THENSLOW:PRINT"ERRORE NELLE LINEE DATA!":END
160 SYS4864:SLOW:SCNCLR:PRINT"COMANDO <SLOAD> IMPLEMENTATO":PRINT:PRINT
170 PRINT"CARICAMENTO:":PRINT:PRINT"DEFAULT  = $8000 (DEC.32768)"
180 PRINT"A SCELTA = POKE5027,HI:POKE5029,LOW":END
190 :
200 DATA 169,016,162,019,141,000,003,142,001,003,096,162,023,076,063,077
210 DATA 224,011,208,249,201,147,208,245,198,061,165,061,201,255,208,002
220 DATA 198,062,032,201,003,201,083,208,228,032,128,003,032,128,003,201
230 DATA 034,208,218,032,128,003,162,000,032,201,003,201,034,240,016,157
240 DATA 063,020,232,224,017,176,196,230,061,208,237,230,062,208,233,134
250 DATA 250,032,069,168,169,015,162,008,168,032,186,255,169,000,032,189
260 DATA 255,032,192,255,173,028,010,041,191,141,028,010,162,015,032,201
270 DATA 255,165,250,105,003,168,162,000,189,060,020,032,210,255,232,136
280 DATA 208,246,032,204,255,044,028,010,112,023,162,000,189,045,020,032
290 DATA 210,255,240,003,232,208,245,169,015,032,195,255,162,255,076,063
300 DATA 077,120,169,128,160,000,132,251,133,252,173,000,221,041,239,141
310 DATA 000,221,032,195,229,173,013,220,032,026,020,201,002,144,036,208
320 DATA 011,169,015,032,195,255,162,004,088,076,063,077,201,031,208,007
330 DATA 032,026,020,170,032,252,019,088,169,015,032,195,255,032,128,003
340 DATA 076,144,175,165,252,201,240,144,011,169,015,032,195,255,162,016
350 DATA 088,076,063,077,032,250,019,076,184,019,162,254,160,000,032,026
360 DATA 020,141,001,255,145,251,169,000,141,000,255,200,202,208,239,152
370 DATA 024,101,251,133,251,144,002,230,252,096,173,000,221,073,016,141
380 DATA 000,221,169,008,044,013,220,240,251,173,012,220,096,068,082,073
390 DATA 086,069,032,078,079,078,032,070,065,083,084,000,085,048,159,000
400 END
```

I CICLI DEL 6502

Determinare la velocità di esecuzione delle singole istruzioni può essere importante; anzi, vitale

di **Giancarlo Mariani**

E' sicuramente noto alla maggioranza che ogni istruzione Basic deve essere convertita in più istruzioni L.M., e che un'istruzione L.M. invece, è direttamente eseguibile dal micro.

Quello che forse è meno noto, è che anche la singola istruzione LM *deve* essere "spezzata" dal micro in tante piccole parti per poter essere eseguita. In pratica, il micro si comporta più o meno come un interprete, che "capisce" le istruzioni LM che gli vengono impartite, e quindi le spezza nelle varie microistruzioni che servono ad eseguire quella stessa operazione.

Il ciclo di esecuzione di una qualsiasi istruzione del 6502 è così composto:

1 - Prelievo (fetch) del codice operativo dell'istruzione dalla memoria.
2 - Decodifica del codice operativo.
3 - Esecuzione dell'istruzione.

I primi due passi saranno uguali per qualsiasi istruzione, mentre il terzo, ovviamente, cambierà a seconda dell'istruzione LM impartita.

Per fare un esempio, l'esecuzione di una LDA (carica nell'accumulatore...) sarà totalmente diversa da quella di un JMP (salta a....). Il terzo passo, ossia l'esecuzione, dovrà essere necessariamente diverso. Va ricordato che questi tre passi sono tutti eseguiti a livello hardware (internamente al microprocessore) perchè fanno parte del suo intimo funzionamento.

Ogni istruzione LM è spezzata dal micro in varie fasi, dette *cicli*. Nella famiglia cui appartiene il microprocessore del C/64 (il famoso 6502) esistono svariati tipi di cicli, tra cui quello di lettura (Read Cycle), quello di scrittura (Write Cycle), quello di attesa (Wait Cycle) e tanti altri.

Ritornando all'esempio di prima, una LDA (in memoria) invocherà, presumibilmente, i seguenti cicli:

1 - Metti l'indirizzo da caricare nell'address bus.
2 - Aspetta che la memoria trasferisca sul data bus il contenuto dell'indirizzo richiesto.
3 - Leggi, dal data bus, il byte proveniente dalla memoria.
4 - Trasferisci il byte nell'accumulatore.

Una JMP (assoluta), invece, potrà essere così composta:

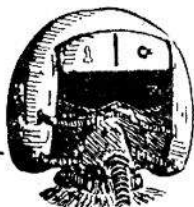
1 - Leggi l'indirizzo specificato
2 - Mettilo in PC (Program Counter)

Si capisce, pertanto, che il micro deve avere al suo interno una specie di "micro interprete" in grado di capire le istruzioni LM e di tradurle nei cicli necessari alla loro esecuzione.

Ad ogni "colpo" di clock viene eseguito un singolo ciclo. Il C/64, che dispone di un clock con frequenza di 1 MHZ, eseguirà 1 milione di cicli al secondo; ne consegue che un ciclo durerà 1 microsecondo. Si può determinare la durata delle singole istruzioni LM.

Qualsiasi manuale del 6502 riporta di solito, di fianco alla spiegazione di ogni istruzione, anche la sua durata in cicli. Ovviamente, più cicli "dura" un'istruzione, più tempo il micro impiegherà ad eseguirla.

Per ritornare all'esempio di prima, una LDA eseguita in modo assoluto "dura" 4 cicli, mentre una JMP (sempre assoluta), solo 3. Questo significa che il 6502, con clock a 1 MHZ, impiegherà 4 microsecondi ad eseguire la LDA e 3 microsecondi per la JMP. Raddoppiando il clock, ovviamente, si dimezza la durata del ciclo e quindi si raddoppia la veloci-

Non tutte le istruzioni sono eguali tra loro

LE AVVENTURE DI

PRIMO GIOVEDINI

"PONG!" (I file)

by Marco Mietta
Barbara De Toffoli

TABELLA DEL TEMPO DI ESECUZIONE DELLE ISTRUZIONI DEL 6502

	Accumulatore	Immediato	Pag.zero	Pag.zero,X	Pag.zero,Y	Assoluto	Assoluto,X	Assoluto,Y	Implicito	Relativo	(Indiretto,X)	(Indiretto),Y	Ass.indiretto
ADC	.	2	3	4	.	4	4+	4+	.	.	6	5+	.
AND	.	2	3	4	.	4	4+	4+	.	.	6	5+	
ASL	2	.	5	6	.	6	7	.	.	.	.	.	.
BCC	.	.	.	.	.	.	.	.	.	2!	.	.	.
BCS	.	.	.	.	.	.	.	.	.	2!	.	.	.
BEQ	.	.	.	.	.	.	.	.	.	2!	.	.	.
BIT	.	.	.	3	.	4	.	.	.	.	.	.	.
BMI	.	.	.	.	.	.	.	.	.	2!	.	.	.
BNE	.	.	.	.	.	.	.	.	.	2!	.	.	.
BPL	.	.	.	.	.	.	.	.	.	2!	.	.	.
BRK	.	.	.	.	.	.	.	.	7	.	.	.	.
BVC	.	.	.	.	.	.	.	.	.	2!	.	.	.
BVS	.	.	.	.	.	.	.	.	.	2!	.	.	.
CLC	.	.	.	.	.	.	.	.	2	.	.	.	.
CLD	.	.	.	.	.	.	.	.	2	.	.	.	.
CLI	.	.	.	.	.	.	.	.	2	.	.	.	.
CLV	.	.	.	.	.	.	.	.	2	.	.	.	.
CMP	.	2	3	4	.	4	4+	4+	.	.	6	5+	.
CPX	.	2	3	.	.	4	.	.	.	.	.	.	.
CPY	.	2	3	.	.	4	.	.	.	.	.	.	.
DEC	.	.	5	6	.	6	7	.	.	.	.	.	.
DEX	.	.	.	.	.	.	.	.	2	.	.	.	.
DEY	.	.	.	.	.	.	.	.	2	.	.	.	.
EOR	.	2	3	4	.	4	4+	4+	.	.	6	5+	.

Durante un ciclo macchina viene eseguita una singola "istruzione" hardware

tà di esecuzione delle istruzioni LM. La relazione per calcolare la durata del ciclo è la solita...

Ciclo = 1 / Frequenza

...dove, se impostiamo la frequenza in MHZ (1 Megahertz = 1000000 Hertz), "Ciclo" resta espresso in microsecondi (1 microsecondo = 1 milionesimo di secondo).

Ovviamente, più complessa è un'istruzione, più numerosi saranno i cicli che occorreranno per eseguirla, e quindi la sua esecuzione risulterà più lenta.

Alcuni microprocessori possiedono delle istruzioni particolari, quali moltiplicazione, divisione ed altre, che vengono eseguite in pochi cicli macchina. Il 6502 non possiede tali istruzioni e, di conseguenza, è necessario scrivere appositi programmi l.m; dovrebbe incominciare ad esser chiaro, insomma, il motivo per cui alcuni micro sono più veloci di altri: a parte una questione di Mhz, infatti, è importante il numero di cicli macchina da effettuare per ottenere un determinato effetto globale.

Esaminando le tabelle delle istruzioni LM dei vari micro esistenti, si scopre, tuttavia, che alcune istruzioni possono durare 20, 30 o addirittura 50 cicli; a volte, paradossalmente, può essere più conveniente, in termini di velocità, scrivere un programmino che simuli l'istruzione desiderata. Questo perchè, facendo il conto dei cicli di durata del programma, ci si rende conto che risulta un minor numero di cicli impiegati dalla singola istruzione. In alcuni casi un intero programma risulta addirittura più veloce di una sola istruzione; lo svantaggio? bè, è chiaro: la maggior occupazione di memoria Ram per ospitare il programma.

RISC CALCOLATO

Ultimamente sono stati introdotti, sul mercato, alcuni microprocessori che usano la tecnica deno-

```
INC     . . 5 6 . 6 7 . . . . . .
INX     . . . . . . . . 2 . . . .
INY     . . . . . . . . 2 . . . .
JMP     . . . . . 3 . . . . . . 5
JSR     . . . . . 6 . . . . . . .
LDA     . 2 3 4 . 4 4+4+. . 6 5+.
LDX     . 2 3 . 4 4 . 4+. . . . .
LDY     . 2 3 4 . 4 4+. . . . . .
LSR     2 . 5 6 . 6 7 . . . . . .
NOP     . . . . . . . . 2 . . . .
ORA     . 2 3 4 . 4 4+4+. . 6 5+.
PHA     . . . . . . . . 3 . . . .
PHP     . . . . . . . . 3 . . . .
PLA     . . . . . . . . 4 . . . .
PLP     . . . . . . . . 4 . . . .
ROL     2 . 5 6 . 6 7 . . . . . .
ROR     2 . 5 6 . 6 7 . . . . . .
RTI     . . . . . . . . 6 . . . .
RTS     . . . . . . . . 6 . . . .
SBC     . 2 3 4 . 4 4+4+. . 6 5+.
SEC     . . . . . . . . 2 . . . .
SED     . . . . . . . . 2 . . . .
SEI     . . . . . . . . 2 . . . .
STA     . . 3 4 . 4 5 5 . . 6 6 .
STX     . . 3 . 4 4 . . . . . . .
STY     . . 3 4 . 4 . . . . . . .
TAX     . . . . . . . . 2 . . . .
TAY     . . . . . . . . 2 . . . .
TSX     . . . . . . . . 2 . . . .
TXA     . . . . . . . . 2 . . . .
TXS     . . . . . . . . 2 . . . .
TYA     . . . . . . . . 2 . . . .
        -------------------------

+ : Aggiungere un ciclo se
l'indicizzazione supera il limite
di pagina.

! : Aggiungere un ciclo se il
salto e' eseguito (condizione
vera). Aggiungere un altro ciclo
se il salto supera il limite di
pagina.
```

minata RISC, che significa "Reduced Instruction Set Computer", ossia "Computer a ridotto numero di istruzioni".

Questi microprocessori sono velocissimi perchè possiedono un numero molto ridotto di istruzioni, le quali però vengono eseguite in 1, 2 o al massimo 3 cicli macchina. Per questa ragione i programmi scritti con questi microprocessori saranno poco comprensibili, data la scarsità e la semplicità delle istruzioni presenti, ma in compenso risulteranno estremamente veloci.

I CICLI SUL C/64

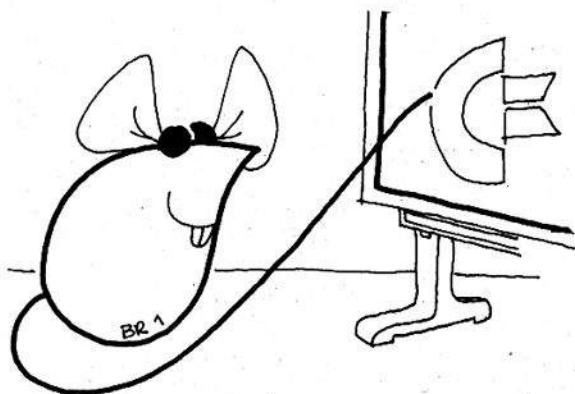
Alla fine di questa chiaccherata, in cui speriamo di aver chiarito i concetti di "ciclo macchina" e di durata delle istruzioni LM, non poteva mancare il tradizionale esempio applicativo, che ha lo scopo di chiarire ulteriormente, in modo pratico, la teoria spiegata nell'articolo.

Il semplicissimo programmino riportato in queste pagine genera un ritardo di 1/10 di secondo. Il lettore può esaminare il disassemblato e, quindi, aiutandosi con la tabella dei cicli, può calcolare il tempo di durata del programma stesso, e verificare che è effettivamente 1/10 di secondo.

Non sono presenti i commenti nel disassemblato perchè è veramente semplicissimo; comunque, per chi avesse difficoltà a capirne il funzionamento, possiamo dire che esso funziona mediante 3 cicli nidificati.

Il primo (più interno) "dura" 400 cicli. Il secondo, ripete 250 volte il primo, quindi 250 x 400 = 100000 cicli. Dato che 1 ciclo = 1 microsecondo, 100000 cicli = 1/10 di secondo.

Esiste anche un terzo ciclo, il più esterno, che permette di ripetere da 1 a 255 volte il tutto, quindi tramite la routine, si possono generare ritardi variabili tra 1/10 di secondo e 25.5 secondi.

Anche un microprocessor e è dotato di un "interprete" al suo interno

Si può notare che l'interrupt mascherabile risulta disabilitato perchè, in caso contrario, verrebbe richiamato 60 volte al secondo generando un ritardo che sfalserebbe l'intero calcolo dei cicli di durata del programma.

Il listato Basic, altrettanto banale, può servire a chiarire ulteriormente le idee.

Non mancheranno sui prossimi numeri di C.C.C. altre applicazioni, decisamente più interessanti e, magari, utili. Nel frattempo sperimentate altre routine e, mi raccomando, occhio al ciclo!

UN MICRO CON ALTI E BASSI

Quando noi accendiamo il nostro home computer Commodore 64, ci troviamo immediatamente in un ambiente ormai a noi familiare, ossia l'*interprete* basic.

Come voi ben sapete, possiamo impartire subito dei comandi al computer, oppure scrivere programmi, farli eseguire e quindi osservarne i risultati.

La parte che si occupa di far *capire* al microprocessore i nostri comandi è il *sistema operativo* (S.O.) del computer, che, assieme all'interprete Basic, costituisce la parte che "traduce", le frasi digitate, in un linguaggio comprensibile al microprocessore, ossia il *linguaggio macchina* (L.M.).

Il *Basic*, come il *Fortran*, il *Cobol*, il *Pascal*, il *C*, e tantissimi altri, sono definiti "linguaggi ad alto livello".

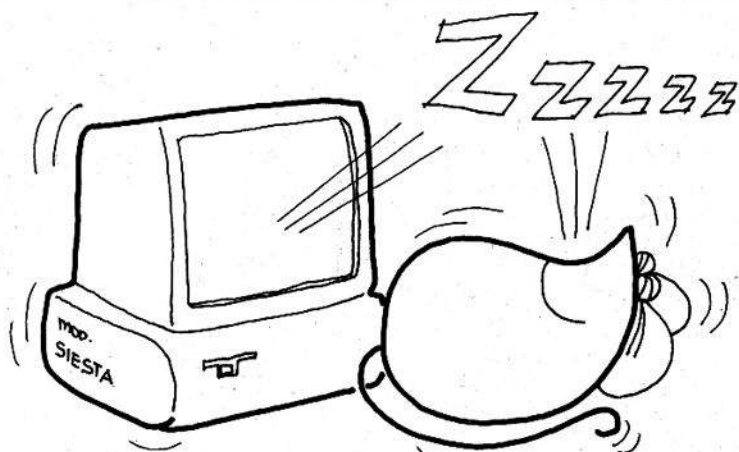
Ciò significa che l'esecuzione di un'istruzione, scritta in uno dei linguaggi sopra citati, corrisponde, in effetti, all'esecuzione di numerose istruzioni in L.M. Ovviamente il L.M. e più precisamente l'Assembler (codifica mnemonica del L.M.) è considerato un linguaggio a "basso livello", dal momento che c'è quasi una corrispondenza diretta di 1 a 1 tra le istruzioni Assembler e le istruzioni L.M.

L'interprete, o il compilatore, presente nella macchina provvederà a "tradurre" le istruzioni scritte in linguaggi ad alto livello, in istruzioni L.M. (ossia in un linguaggio a basso livello).

I linguaggi ad alto livello sono stati ideati per *semplificare* la programmazione: grazie a loro, infatti, un qualsiasi programma può essere scritto usando comuni termini inglesi, comprensibili da tutti(!) noi, quindi, con la massima semplicità.

I linguaggi, inoltre, hanno il vantaggio di non dipendere dal microprocessore usato. Un listato in Basic scritto per un computer basato sul microprocessore 6502, ad esempio, può fornire gli stessi risultati su un elaboratore costruito "attorno" al micro Z-80 o a qualsiasi altro processore.

Al contrario, il L.M. presume uno studio più o meno approfondito del microprocessore con cui si opera, ed i programmi scritti in questo linguaggio, se non bastasse, risulteranno essere poco chiari ai meno esperti.

Riassumendo: il microprocessore "capisce" solo il L.M, ogni processore ha il suo L.M. quindi esisterà un L.M. 6502, un L.M. Z-80, un L.M. 68000, e così via. I linguaggi ad alto livello, quali il Basic o il Pascal, non sono direttamente capiti dal processore, ma devono convertire ogni loro istruzione in una sequenza di istruzioni L.M. per essere direttamente eseguibili dal micro.

```
Disassemblato
routine di ritardo

By Mariani G.

Delay :   SEI
Loop  :   LDA #$EA
          STA $FB
Loop1 :   LDA #$32
          STA $FC
Loop2 :   DEC $FC
          BNE Loop2
          DEC $FB
          BNE Loop1
          DEC $02
          BNE Loop
          CLI
          RTS
```

```
10 REM ** APPLICAZIONE CICLI MACCHINA
20 REM ** BY MARIANI G.
30 :
40 REM ** USO:
50 REM ** POKE 2,N:SYS X
55 :
60 REM ** N= NUMERO DI DECIMI DI SEC.
70 REM ** X= INDIRIZZO DI PARTENZA
80 :
85 X=36864 : REM ** RILOCABILE
90 :
95 PRINT CHR$(147)
100 S=0
110 FOR T=0 TO 22:READ A:POKE X+T,A:S=S+A:NEXT T
120 IF S<>4150 THEN PRINT:PRINT"ERRORE NEI DATI LM!!!":END
130 PRINT CHR$(147):PRINT:PRINT"QUANTI 10' DI SEC. VUOI ASPETTARE?"
140 INPUT D
150 PRINT:PRINT"ASPETTA";D/10;"SECONDI..."
160 POKE 2,D:SYS X
1000 :
1010 REM ** DATI ROUTINE LM **
1020 :
1100 DATA 120,169,234,133,251,169,050
1110 DATA 133,252,198,252,208,252,198
1120 DATA 251,208,244,198,002,208,236
1130 DATA 088,096
```

UN SOLO CERVELLO PER MILLE FUNZIONI

Il C/64, grazie ad una semplicissima procedura, è in grado di emulare una delle caratteristiche più interessanti dell'Amiga: la multiprogrammazione

di **Giancarlo Mariani**

Con il C/64 è possibile emulare le notevoli potenzialità di Amiga

UN PO' DI TEORIA

La parola inglese **interrupt**, come facilmente intuibile, significa **interruzione**. Questo termine si addice perfettamente alla funzione che essa svolge perchè questa è, appunto, l'interruzione di una elaborazione allo scopo di ottenere risultati particolari.

In termini informatici, quando viene attivato un interrupt, il programma principale viene interrotto, per svolgerne un altro, al termine del quale verrà ripreso il programma principale esattamente dal punto in cui era stato lasciato.

Per capire meglio occorre fare un esempio della vita di tutti i giorni. Facciamo l'ipotesi che voi siate seduti in poltrona e leggete un giornale; il vostro **programma principale** è quindi quello di leggere il giornale.

Se, ad un certo punto, squilla il telefono (= interrupt) voi interrompete la lettura del giornale, per rispondere all'apparecchio. Una volta eseguita la routine di interrupt (terminata la conversazione), ritornate al programma principale (vale a dire la lettura del giornale) esattamente dal punto in cui vi eravate fermati.

In termini informatici, significa che mentre "gira" un programma, può avvenire una richiesta di interrupt, e quindi il computer "molla" il programma che stava eseguendo per svolgere quello in interrupt; solo al suo termine ritorna al programma principale. Lo scopo di una tale tecnica verrà descritto in seguito.

Esistono principalmente due tipi di interrupt: il primo è definito **Interrupt non mascherabile** (NMI = Non Maskable Interrupt), mentre il secondo **Interrupt mascherabile** (INT = Interrupt). Vediamo la differenza.

Interrupt non mascherabile, come si nota dalla parola, significa che non può essere ignorato; ne consegue che il microprocessore, se riceve un segnale di questo tipo, DEVE soddisfare la richiesta di interrupt; non può fingere che non sia arrivata. Nell'analogia di prima, appena squilla il telefono voi DOVETE alzarvi, e quindi rispondere all'apparecchio.

L'interrupt mascherabile, invece, tramite un opportuno "interruttore" software, può essere ignorato dal micro; ritornando all'esempio di prima, l'interruttore potrebbero essere costituito da tappi nelle orecchie, che non fanno sentire lo squillo. Allo stesso modo, si può imporre, a un microprocessore, di ignorare la richiesta di interrupt mascherabile, ma NON si può evitare l'esecuzione del NMI.

IL 6502 / 6510

I microprocessori che interessano più da vicino, ossia il 6502 (Vic 20) e il 6510 (C/64) hanno, tra i 40 pin, un piedino denominato IRQ (Interrupt Request, ossia Richiesta di Interruzione). Quando su questo piedino (il n. 3 per il 6510 ed il n. 4 per il 6502) è presente un segnale di livello appropriato (livello logico 0), il microprocessore "molla" tutto quello che stava facendo e compie le seguenti operazioni:

UNA QUESTIONE DI VELOCITA'

Molti, anzi, moltissimi anni or sono, gli unici strumenti che permettevano di svolgere semplici calcoli erano... le 10 dita delle mani (ed è per questo che ancora oggi contiamo in multipli di 10). Un po' più tardi fu inventato l'abaco, che permise di semplificare e velocizzare i calcoli.

In seguito, la storia dell'automazione del calcolo proseguì attraverso l'invenzione di strani congegni meccanici, più o meno complicati, fino a giungere all'ormai lontanissimo 1930, quando fu costruito il primo computer **analogico**(!), che riusciva a calcolare in modo relativamente veloce lunghe liste di problemi matematici.

Già nel 1950 iniziava la prima generazione di computer, basati su tubi a vuoto (le valvole, per intenderci), che potevano compiere qualche migliaio di operazioni al secondo. La tecnica andò sempre più perfezionandosi finchè, verso il 1960, nacque la seconda generazione di computer che, usando **transistor** al posto dei tubi, aveva una velocità di elaborazione nettamente superiore a quella dei predecessori.

Finalmente, nel 1965, comparvero gli ormai famosissimi circuiti integrati, che permisero di costruire computer molto più piccoli e, soprattutto, più veloci (circa 1000 volte più veloci degli elaboratori della prima generazione).

Le macchine dei giorni nostri sono i **Computer della quarta generazione** (anche se, ormai, stiamo per entrare nella quinta...) e sono diventati di una compattezza, di un'affidabilità, di una potenza e di una velocità incredibili rispetto ai loro predecessori di soli 10 anni fa.

Il breve cenno di storia è stato utile per evidenziare l'enorme sviluppo dei computer in pochi anni, soprattutto in termini di velocità di elaborazione. E' quest'ultima, infatti, che raggiungendo nei più recenti computer livelli elevatissimi, ha permesso di sviluppare interessanti applicazioni, quali la multiprogrammazione o il time-sharing, che sono appunto gli argomenti principali del presente articolo.

Risulta quindi doverosa una breve panoramica su di un "animale informatico", che permette lo svolgimento di varie funzioni, e che dovrebbe essere ormai conosciuto dalla stragrande maggioranza di utenti di computer, ossia l'**interrupt**.

Numerosissime sono le riviste del settore, tra cui spicca naturalmente CCC, che più volte hanno trattato l'importante argomento che rappresenta una delle più valide risorse dei computer, siano essi mainframe oppure piccoli home. Chi non sapesse ancora di che cosa si sta parlando, ecco una spiegazione completa (con particolari riferimenti al C/64), che speriamo possa essere utile per chiarire le idee a tutti.

- Termina l'istruzione in Linguaggio Macchina (e **non** Basic!) che stava eseguendo al momento dell'arrivo dell'interruzione.
- Memorizza il contatore di programma (PC)
- Salva il registro di stato (Flags)
- Salta ad eseguire il programma in LM, il cui indirizzo di partenza è contenuto nelle locazioni FFFE (parte bassa) e FFFF (parte alta).
- Al termine della routine di interrupt, il micro ripristinerà PC e Flags e ritornerà ad eseguire ciò che stava facendo prima della richiesta di interruzione.

Tutte le operazioni descritte sono di tipo Hardware, e dipendono strettamente dal microprocessore preso in considerazione, nel nostro caso il 6502/10; tutti gli altri processori, comunque, hanno un comportamento molto simile.

INTERRUPT, ISTRUZIONI PER L'USO

Finora abbiamo parlato di interrupt, di come viene gestito dal 6502, che operazioni esegue; ma.... a che serve?

Manipolando l'interrupt è possibile far eseguire fino ad otto routine contemporanea mente

```
10 REM ****************************
12 REM * PROGRAMMA PER FAR GIRARE *
14 REM * CONTEMPORANEAMENTE OTTO  *
16 REM * ROUTINES LM IN INTERRUPT *
18 REM *  BY MARIANI GIANCARLO    *
20 REM *                          *
22 REM * USO: SYS XXXX            *
24 REM *      (XXXX- IND.PARTENZA)*
25 REM *                          *
26 REM * $FB: OGNI BIT DI QUESTA  *
28 REM *      LOCAZIONE ABILITA   *
30 REM *      UNA ROUTINE (QUANDO *
32 REM *      E' A 1)             *
36 REM ****************************
40 :
41 REM ****************************
42 REM * X-INDIRIZZO DI PARTENZA  *
44 REM * RILOCABILE SOLO TRAMITE  *
46 REM * QUESTO PROGRAMMA         *
48 REM ****************************
50 :
60 X=49152
65 UX$-"ERRORE DI TRASCRIZIONE DATI!"
80 PRINT CHR$(147)
82 PRINT:PRINT" USO: SYS XXXX"
83 PRINT"(SYS"X"IN QUESTA VERSIONE)"
84 PRINT:PRINT " POI, MODIFICARE LA "
85 PRINT"LOCAZIONE 251 COME DESIDERATO"
100 S=0
105 FOR T-0 TO 63
110 READ A
120 POKE X+T,A:S-S+A
130 NEXT T
140 IF S<>6621 THEN PRINT:PRINT UX$:END
150 B-X+19:GOSUB 800
160 POKE X+8,BB:POKE X+13,BA
161 B-X+56:GOSUB 800
162 POKE X+29,BB:POKE X+30,BA
170 :
180 REM * LETTURA IND. DI PARTENZA
190 :
200 FOR T-0 TO 7
210 READ B:GOSUB 800
220 POKE 826+(T*2),BB:POKE 827+(T*2),BA
230 NEXT T
240 :
250 REM * LETTURA ROUTINES IN INTERRUPT
260 :
270 S-0
280 FOR T-0 TO 86
290 READ A
300 POKE 49216+T,A:S-S+A
310 NEXT T
320 IFS<>10001THEN PRINTUX$"ROUT.INTERRUPT!":END
798 LIST-60:     END
799 :
800 REM ** TRASFORMA UN NUMERO IN **
805 REM ** BYTE ALTO/BYTE BASSO   **
810 :
815 BA=INT(B/256):BB-B-BA*256
820 RETURN
1091 :
1092 REM  ****************************
1094 REM  *ROUTINE PER ABILITAZIONE *
1095 REM  *INTERRUPT                *
1098 REM  ****************************
1099 :
1100 DATA 169,000,133,252,133,251,120
1110 DATA 169,023,141,020,003,169,192
1119 DATA 141,021,003,088,096
1121 :
1122 REM  ****************************
1124 REM  * ROUTINE DI ASSERVIMENTO *
1125 REM  * INTERRUPT               *
1128 REM  ****************************
1129 :
1130 DATA 230,252,165,252,041,007,133
1140 DATA 252,170,189,060,192,037,251
1150 DATA 208,003,076,049,234,138,010
1160 DATA 170,189,058,003,141,074,003
1170 DATA 189,059,003,141,075,003,108
1180 DATA 074,003,001,002,004,008,016
1190 DATA 032,064,128
1201 :
1202 REM  ****************************
1204 REM  * INDIRIZZI DI PARTENZA    *
1205 REM  * DELLE 8 ROUTINES MESSE   *
1206 REM  * IN INTERRUPT             *
1208 REM  ****************************
1209 :
1210 DATA 49216,49222,49228,49234
1220 DATA 49252,49258,49272,49290
61000 :
61010 REM  ****************************
61020 REM  * ROUTINES IN INTERRUPT    *
61030 REM  * (PARTENZA-49216)         *
61040 REM  ****************************
61050 :
61100 DATA 238,032,208,076,049,234,238
61110 DATA 033,208,076,049,234,238,134
61120 DATA 002,076,049,234,166,002,169
61130 DATA 032,157,000,004,232,134,002
61140 DATA 169,042,157,000,004,076,049
61150 DATA 234,238,231,007,076,049,234
61160 DATA 165,254,041,015,073,015,133
61170 DATA 254,141,024,212,076,049,234
61180 DATA 166,255,169,032,157,230,006
61190 DATA 202,134,255,169,083,157,230
61200 DATA 006,076,049,234,173,000,006
61210 DATA 041,033,073,001,141,000,006
61220 DATA 076,049,234
61230 END
```

Presteremo, per ora, una particolare attenzione solo sul tipo di interrupt mascherabile, dato che l'NMI verrà trattato prossimamente, in un articolo a parte.

Ricordiamo che i riferimenti contenuti in questo articolo fanno capo soprattutto al C/64, anche se i principi generali sono validi per la maggior parte dei computer in circolazione.

Quando noi accendiamo il 64, questo sembra inattivo, ma il suo cuore, il 6510, pulsa incessantemente elaborando migliaia di istruzioni ogni secondo. Ciò può essere reso immediatamente visibile anche dal semplice lampeggio del familiare rettangolino, il cursore.

Al piedino IRQ del micro è collegato un clock, ossia un orologio, che ogni sessantesimo di secondo (quindi, 60 volte al secondo) fa "scattare" il livello logico zero sul menzionato pin. Il 6502, avvertendo la richiesta, interrompe l'elaborazione ed esegue tutte le operazioni descritte prima, tra le quali, alla fine, è sempre presente il salto indiretto a FFFE.

A questa locazione (ed alla successiva FFFF) è presente l'indirizzo di un programma contenuto nella ROM del computer, che, eseguito 60 volte al secondo, permette di controllare una lunga sequenza di funzioni, quali:

- Lampeggio del cursore.
- Controllo ed eventuale visualizzazione dei tasti premuti.
- Aggiornamento dell'orologio di sistema (variabili TI e TI$)
- Controllo dello scambio di dati tra computer e periferiche.
- Altre funzioni particolari.

E' possibile accorgersi che, effettivamente, ci sia un "qualcosa" che le esegue in modo trasparente dal programma utente, se noi, in qualsiasi momento, schiacciamo un tasto: questo, infatti, viene immediatamente riconosciuto e visualizzato. Ciò dimostra che deve esser presente una routine che "veglia" costantemente sulla tastiera.

Allo stesso modo, deve esistere anche una routine che costantemente aggiorna l'orologio: provando a digitare il semplice programmino...

```
10 PRINT CHR$ (147); TI$
20 GOTO 10
```

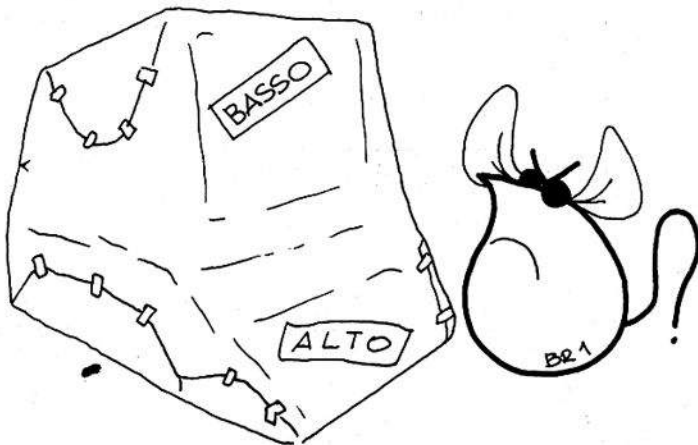
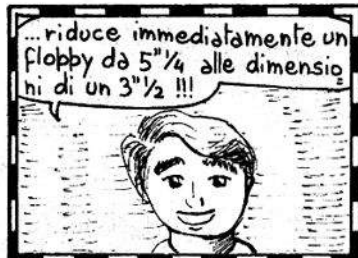
Le routine in interrupt, in pratica, sembra che giri-

```
Start: SEI            ;Disabilita interrupt
       LDA #$00       ;Ind.LO routine utente
       STA $0314      ;Mette in 0314
       LDA #$C0       ;Ind.HI routine utente
       STA $0315      ;Mette in 0315
       CLI            ;Ripristina interrupt
       RTS            ;Ritorna a prog.princ.
```

Figura 1

```
$C000: INC $D020 ;Incr. colore di bordo
       JMP $EA31 ;Salta a norm.interr.
```

Figura 2

```
NewInt:    Chiama programma X
           Incrementa X
           X e' > del n. programmi?
           Si': Resetta X
           Chiama interrupt S.O. ($EA31)
```

Figura 3

no contemporaneamente al programma principale, ma in realtà non è così, anche se, essendo eseguite 60 volte al secondo, possono dare l'impressione di essere sempre attive.

Si intuisce che se potessimo sfruttare in qualche modo la tecnica di interrupt, potremmo creare applicazioni interessanti.

Sia le routine in interrupt, sia le locazioni FFFE e FFFF (che contengono l'indirizzo di partenza di esse) sono contenute in ROM, e quindi non modificabili. Come sarà possibile, allora, sfruttare questa tecnica?

Esaminando le locazioni FFFE e FFFF, si nota che contengono, rispettivamente, i valori esadecimali 48 e FF. Ciò significa che ogni volta che al 6502 arriva una richiesta di interruzione, questo esegue la routine posta a partire dall'indirizzo FF48. Disassemblandola, si nota che ad un certo punto, (quasi subito) esegue un salto indiretto alla locazione esadecimale 0314.

Guarda caso! 0314 (decimale 788) è in RAM, quindi modificabile con la massima facilità. Questa locazione, insieme alla 0315, contiene un indirizzo, (EA31) che è quello di partenza della routine vera e propria di interrupt mascherabile. Ciò vuol dire che la normale routine di interrupt mascherabile del C/64 inizia dall'indirizzo esadecimale EA31, pur se la procedura è costretta a "passare" attraverso le locazioni 0314 - 0315.

Manipolando opportunamente l'indirizzo contenuto nelle locazioni 0314 - 0315, possiamo quindi far eseguire al computer una qualsiasi **nostra** routine (purchè scritta in LM) 60 volte al secondo, in modo completamente trasparente al programma che sta girando.

Si può immaginare quali possibilità apra questa tecnica.

LE ISTRUZIONI DELL'INTERRUPT

Vi sono tre istruzioni che permettono di gestire l'interrupt attraverso l'assembler del 6502: SEI, CLI e RTI.

SEI (Set Interrupt Disable) impone al 6502 di ignorare, da quel momento in poi, ogni richiesta di interrupt mascherabile. In pratica, ha la stessa funzione dei tappi nelle orecchie dell'esempio di prima.

CLI (Clear Interrupt Disable) è l'esatto contrario di SEI, ossia toglie i tappi, permettendo al 6502 di sentire e quindi eseguire le richieste di interrupt.

RTI (ReTurn from Interrupt) deve essere inserita alla fine della routine di asservimento dell'interrupt (ad esempio, quella che inizia a EA31), e comunica al 6502 che la routine di interrupt è finita e che il micro può ritornare ad eseguire il programma principale, ripristinando i vari registri e flags salvati prima.

E' da notare che esiste una certa analogia tra le chiamate di subroutine tramite istruzioni JSR / RTS e le chiamate di routines in interrupt (RTI), anche se lo scopo è completamente diverso.

SEMPLICI PROGRAMMI IN INTERRUPT

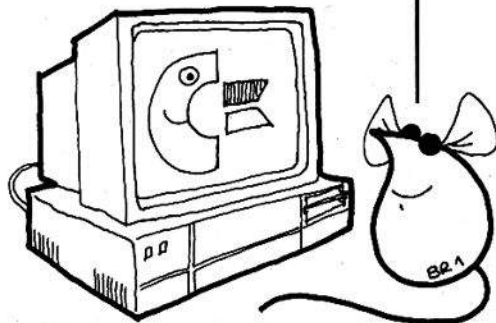
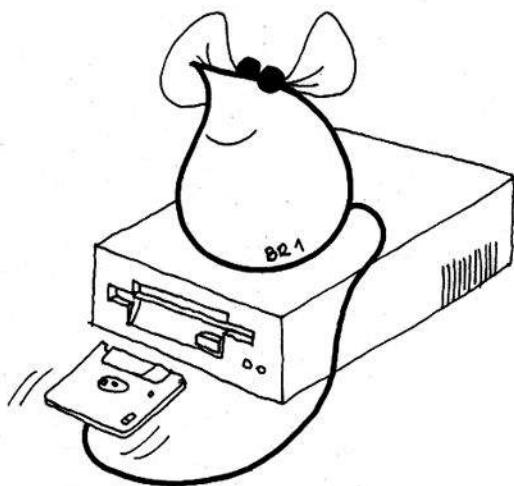
Se quanto letto sinora è stato correttamente appreso, si può intuire che per inserire una nostra routine nel ciclo di interrupt, per farla eseguire 60 volte al secondo, è sufficiente cambiare il contenuto delle locazioni 0314 - 0315, inserendo in queste, rispettivamente, la parte bassa e la parte alta dell'indirizzo di partenza della routine stessa.

Supponiamo che questa parta da $C000 (decimale 49152) e che, nella sua voluta banalità, si limiti a cambiare il colore del bordo dello schermo.

Ammesso di aver digitato la routine, bisogna ora cambiare il contenuto delle locazioni 0314 - 0315, facendole "puntare" non più a EA31, ma a C000. Questo può essere ottenuto mediante semplici istruzioni LDA - STA, ma **attenzione**! non sempre è una procedura corretta!

Ricordiamo, infatti, che la routine di interrupt viene chiamata molto frequentemente (60 volte in un secondo); se, quindi, modifichiamo l'indirizzo in maniera tanto disinvolta, può accadere che venga effettuata una richiesta di interrupt proprio nel momento in cui abbiamo già modificato la parte alta dell'indirizzo e ci stiamo apprestando a modificarne la parte bassa (o viceversa). Il 64 si ritroverebbe un indirizzo modificato a metà e salterebbe in un'area di memoria, dove probabilmente non è contenuto niente, "inchiodandosi" senza possibilità di ritorno, se non spegnendo.

Prima di cambiare gli indirizzi, quindi, bisogna dire al 6502 di ignorare, da quel momento in poi, ogni richiesta di interrupt; quindi cambiare l'indiriz-

```
MULTIPROGRAMMAZIONE CON IL C/64
DISASSEMBLATO COMMENTATO


;* Routine per spostare i vettori di interrupt *

Init: LDA #0
      STA $FC     ;Contatore routines
      STA $FB     ;Maschera x routines
      SEI         ;Disabilita interrupt
      LDA #<Gest ;Indir. LO rout.gest.
      STA $0314   ;Vettore int. LO
      LDA #>Gest ;Indir. HI rout.gest.
      STA $0315   ;Vettore int. HI
      CLI         ;Abilita interrupt
      RTS         ;Ritorna al Basic

;* Routine di gestione dell'interrupt, puntata da $0314-0315 *

Gest: INC $FC     ;Incrementa contatore
      LDA $FC     ;Lo maschera con 7
      AND #7      ;(va da 0 a 7 =
      STA $FC     ; 8 routines)
      TAX         ;Trasferisce in X
      LDA Mask,X ;Carica masch. da tab.
      AND $FB     ;Seleziona bit
      BNE Call    ;Se e' a 1,chiama rout
      JMP $EA31   ;Se no, interrupt S.O.

Call: TXA         ;Carica indirizzo
      ASL         ;di partenza della
      TAX         ;routine da richiamare
      LDA $033A,X;da tabella indirizzi.
      STA $034A   ;(posta a $033A)
      LDA $033B,X;Parte alta ind.
      STA $034B
      JMP ($034A);ESEGUE ROUTINE
```

```
;* Tabella bytes per mascherare ogni bit della loc. 251 *

Mask: .BYTE $01,$02,$04,$08,$10,$20,$40,$80


;* Tabella indirizzi di partenza routines da richiamare in interrupt

$033A:.WORD $C040,$C046,$C04C,$C052
$0342:.WORD $C064,$C06A,$C078,$C08A

;* Routines di esempio *


;1-Incrementa colore bordo

$C040: INC $D020
       JMP $EA31


;2-Incrementa colore sfondo

$C046: INC $D021
       JMP $EA31


;3-Incrementa colore carattere

$C04C: INC $0286
       JMP $EA31


;4- "*" che viaggia sullo schermo

$C052: LDX $02       ;Punt. loc. schermo
       LDA #$20      ;Spazio
       STA $0400,X
       INX
       STX $02
```

```
         LDA #$2A      ;Asterisco
         STA $0400,X ;Scrive *
         JMP $EA31

;5-Modifica ultima locazione schermo

$C064:   INC $07E7     ;2023 (ultima loc)
         JMP $EA31

;6-Effetto Sonoro

$C06A:   LDA $FE
         AND #15
         EOR #15
         STA $FE
         STA $D418     ;Controllo volume
         JMP $EA31

;7-Cuoricino che viaggia sullo schermo

$C078:   LDX $FF
         LDA #$20
         STA $06E6,X
         DEX
         STX $FF
         LDA #$53
         STA $06E6,X
         JMP $EA31

;8-Punto esclamativo che lampeggia

$C08A:   LDA $0600
         AND #$21
         EOR #1
         STA $0600
         JMP $EA31

; (Fine)
```

zo, ed infine ripristinare il normale asservimento dell'interrupt.

Un programmino per eseguire in interrupt una nostra routine posta a partire da $C000 potrebbe essere quello di figura 1.

A partire da $C000 va, ovviamente, inserita la nostra routine, che, come detto prima, servirà solo a cambiare colore allo schermo (figura 2). Tale routine DEVE esser già presente al momento della prima attivazione del dirottamento dell'interrupt.

La prima istruzione (di figura 2) incrementa il colore del bordo. Alla fine della nostra routine, poi, è assolutamente necessario saltare al normale asservimento dell'interrupt del 64, altrimenti non verrebbe più gestita, tra l'altro, la tastiera ed il 64 si inchioderebbe inesorabilmente. Per evitare ciò è presente la seconda istruzione del programma di figura 2, che ripristina le normali condizioni di funzionamento.

Per far partire il nostro programmino è sufficiente impartire **SYS Start**. Se non avete commesso errori, il colore del bordo continuerà a cambiare, mentre voi potrete fare qualsiasi cosa, come digitare comandi basic, eseguire listati, e così via.

In pratica, ora è anche la nostra routine (posta a $C000) ad essere eseguita 60 volte al secondo, richiamando poi anche il normale interrupt. Questa tecnica, ad esempio, viene usata per molti giochi, dove la musica "gira" in interrupt, mentre il gioco si svolge per conto proprio, oppure in altri programmi particolari, che tutti voi sicuramente avrete avuto modo di vedere.

Alcuni accorgimenti devono essere usati per scrivere routines in interrupt, onde evitare malfunzionamenti ed "inchiodamenti" di vario tipo.

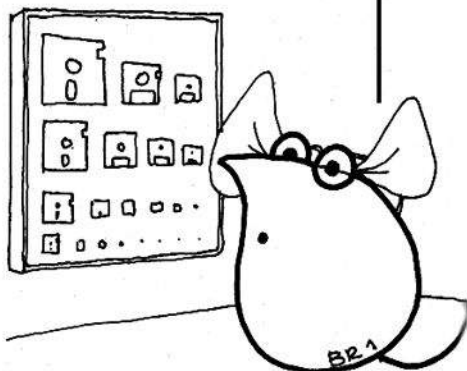
- Come visto prima, la "nostra" routine in interrupt DEVE terminare con un JMP $EA31, per consentire il normale svolgimento delle funzioni di interrupt del S.O.
- Questa deve essere BREVE, non tanto come lunghezza, ma come tempo di esecuzione. Questo perchè il 64 richiama la routine 60 volte al secondo. Se questa durasse più di 1/60 di secondo, verrebbe richiamata di nuovo mentre la si sta eseguendo, e così via per un gran numero di volte, riempiendo lo stack e inchiodando il 64. Quindi, routines PIU' BREVI di 1/60 di secondo.
- Nel caso in cui la nostra routine di interrupt usi registri quali A, X oppure Y, è buona norma salvarli, e quindi ripristinarli prima di eseguire il JMP $EA31,

TABELLE (MULTIPROGRAMMAZIONE SUL 64)

Tab. 1: Allocazione delle 8 routines di esempio

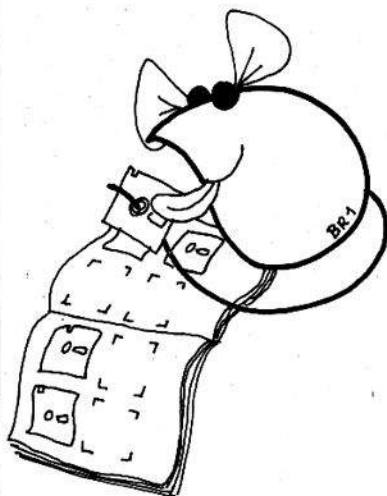
```
1: $C040 - $C045 = Colore Bordo
2: $C046 - $C04B = Colore Sfondo
3: $C04C - $C051 = Colore Caratteri
4: $C052 - $C063 = * che viaggia
5: $C064 - $C069 = Loc. schermo che cambia
6: $C06A - $C077 = Effetto sonoro
7: $C078 - $C089 = Cuoricino che viaggia
8: $C08A - $C096 = ! lampeggiante
```

Tab 2: Allocazione routine di gestione interrupt (rilocabile tramite programma basic)

```
$C000 - $C012 : Modifica vettori int.
$C013 - $C037 : Routine di gestione
$C038 - $C03F : Tabella maschera bit
```

Tab.3: Tabella indirizzi di partenza delle 8 routines

```
$033A - $033B : Prima routine    ($C040)
$033C - $033D : Seconda routine  ($C046)
$033E - $033F : Terza routine    ($C04C)
$0340 - $0341 : Quarta routine   ($C052)
$0342 - $0343 : Quinta routine   ($C064)
$0344 - $0345 : Sesta routine    ($C06A)
$0346 - $0347 : Settima routine  ($C078)
$0348 - $0349 : Ottava routine   ($C08A)
```

Tab 4: Locazioni usate dal programma:

```
$034A - $034B : Indirizzo routine
                da chiamare
$FB : Seleziona le varie routines
      da chiamare
$FC : Contatore di routines (0 - 7)
$02, $FE, $FF : Usati dalle routines
                di esempio
```

anche se in realtà questa operazione non è necessaria, dal momento che ci pensa già il S.O.

Terminata la lunga panoramica sull'interrupt, si può finalmente passare a parlare di multiprogrammazione, e sul modo di simularla con il 64.

LA MULTIPROGRAMMAZIONE

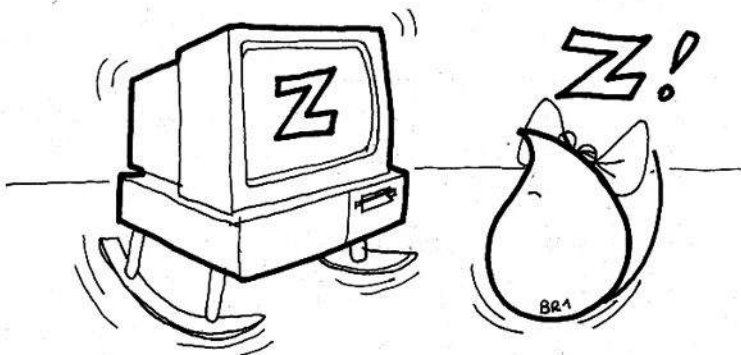
La multiprogrammazione è una applicazione particolare dell'interrupt, che consente a più programmi di girare contemporaneamente sullo stesso calcolatore. Per fare un esempio, riferiamoci sempre alla macchina più perfetta che sia mai stata creata, cioè l'Uomo.

In questo momento state leggendo la rivista, ma non solo! In questi stessi istanti state respirando, il vostro cuore batte, le vostre mani sorreggono il giornale nella corretta posizione, gli occhi seguono il movimento giusto per consentire la lettura, e tante altre cose. Tutte queste attività possono essere considerate veri e propri programmi, fatti girare contemporaneamente dalla vostra CPU, che è poi il cervello.

Alla stessa maniera, su di un solo computer possono essere fatti eseguire più programmi contemporaneamente, o quasi, ed il numero e la complessità di questi dipendono soltanto dalla memoria e dalla velocità dell'elaboratore in questione.

A pensarci bene, nel senso stretto del termine si è sempre in caso di multiprogrammazione, dato che le routines in interrupt girano costantemente assieme al programma utente. Noi, comunque, definiremo multiprogrammazione quell'attività che consente a due o più **programmi utente** di girare nello stesso momento.

MULTIPROGRAMMAZIONE, COME OTTENERLA

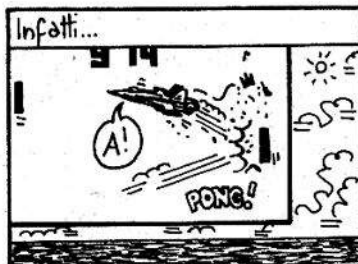
Come detto prima, la multiprogrammazione è una applicazione dell'interrupt, e da ciò si capisce che i vari programmi utente da eseguire contemporaneamente si devono richiamare in interrupt. Facciamo un esempio:
Abbiamo detto che il 64 ogni 60' di secondo richiama particolari routines S.O. e come sia possibile modificare alcuni vettori che consentano di saltare, invece, ad una nostra routine particolare, che magari al primo 60' di secondo richiama un programma, al secondo 60' un altro, al terzo un altro, e così via.

In pratica, la nostra nuova routine in interrupt, ad ogni ciclo deve richiamare un programma diverso, fino ad esaurire questi programmi e quindi ricominciare da capo. Il diagramma di una simile routine potrebbe essere quello di figura 3.

Ovviamente i programmi utente da far girare contemporaneamente devono essere anch'essi routine in interrupt, e quindi devono avere le caratteristiche specificate prima, quali durata breve, chiamata a $EA31, e così via.

Da ciò si capisce che i vari programmi utente NON girano effettivamente assieme, ma, dal momento che questi vengono richiamati parecchie volte al secondo, l'operatore effettivamente non si accorge che la CPU ne elabora uno alla volta, in rapidissima sequenza. Data la relativa lentezza del 64, comunque, è possibile far girare assieme solo un numero limitato di routines e queste devono essere anche abbastanza semplici e brevi.

Un buon esempio di multiprogrammazione possiamo averlo osservando l'orologio dell'AMIGA. Questo, infatti, può essere richiamato un gran numero di volte, e sullo schermo si vedranno tanti orologi che sembrano segnare la stessa ora. Osservando attentamente, tuttavia, ci si può accorgere del leggero sfasamento tra la posizione delle lancette dell'orologio chiamato per primo e quello chiamato per ultimo. Questo dimostra che la CPU elabora sequenzialmente, uno alla volta, tutti i programmi richiamati, ma, data la sua velocità, sembra che questi funzionino assieme.

Il **Time Sharing** (condivisione di tempo) è un'altra applicazione dell'interrupt. In questo caso vi so-

Non c'è limite teorico al numero di programmi che possono essere eseguiti in Interrupt

Ciascun bit della locazione 251 "comanda" una delle otto routine possibili

no vari terminali collegati ad una stessa CPU, ed è quest'ultima che in interrupt, ciclicamente, va ad "ascoltare" prima il terminale n.1, poi il n.2, il n.3, eccetera. In pratica, nella multiprogrammazione la CPU divide il suo tempo tra molti programmi su di un solo terminale, mentre nel time-sharing lo divide tra molti terminali.

UN ESEMPIO SUL 64

Come applicazione di quanto spiegato sinora, ho costruito un semplice programma che permette di far girare contemporaneamente in interrupt fino a 8 brevi routines, che potranno svolgere compiti completamente diversi. Seguendo attentamente la spiegazione, l'utente sarà in grado alla fine, sfruttando questo programma, di creare proprie routines LM e quindi di inserirle in propri programmi.

La tecnica usata dal programma di queste pagine è quella spiegata in precedenza.

• Vengono modificati i vettori di interrupt ($0314-$0315), spostandoli in un punto dove è presente la routine di gestione vera e propria.

• Questa, tramite un contatore, richiama ogni volta che viene attivata (quindi, essendo in interrupt, ogni 60' di secondo) un programma LM diverso, il cui indirizzo di partenza è contenuto in una tabella che è "puntata" appunto dal contatore.

Per come è strutturato il programma, è possibile richiamare fino a 8 routines diverse in interrupt, quindi ognuna di queste sarà eseguita ogni 8/60 di secondo. Esaminando il listato basic ed i relativi disassemblati LM si può capire come funziona il programma. In ogni caso, per personalizzare la procedura, si può cambiare sia l'indirizzo di partenza della routine di gestione che quelli delle routines da far girare in interrupt. Bisogna fare comunque attenzione perchè, dato che la routine di gestione dell'interrupt NON è rilocabile, il suo indirizzo si può cambiare solo tramite il programma pubblicato (come si legge nel listato), che provvede a modificare alcuni bytes necessari al suo funzionamento.

Per far girare l'esempio, è sufficiente digitare il programma Basic, salvarlo(!), dare RUN e quindi, ammesso che non segnali errori, impartire SYS X, dove X è l'indirizzo di partenza della routine (nell'esempio, 49152).

Apparentemente sembra che non succeda nulla; in pratica, però, avete sotto controllo ben 8 programmi, che potete far girare contemporaneamente in interrupt. Come esempio dimostrativo sono state inserite otto semplicissime routines, che sono:

Bit (locazione 251)

0 - Incrementa colore Bordo
1 - Incrementa colore Sfondo
2 - Incrementa colore caratteri
3 - Asterisco che si muove sullo schermo
4 - Cambia ultima casella dello schermo
5 - Effetto sonoro
6 - Cuoricino che si muove sullo schermo
7 - Punto esclamativo lampeggiante

Tutte le routines (volutamente banali per garantire la brevità di digitazione) vengono controllate tramite la locazione esadecimale $FB (decimale 251). Ogni bit di questa locazione controlla la corrispondente routine: quando è posto a zero, la routine non viene richiamata, mentre quando è settato a 1 viene lanciata la sua esecuzione nel ciclo di interrupt.

Provate ad inserire dei valori in questa locazione (ovviamente dopo aver impartito la SYS prima indicata...) e vedrete l'effetto. Con POKE 251, 255 tutte le otto routines saranno richiamate in interrupt ciclicamente, e quindi sembreranno girare contemporaneamente. In pratica potrete digitare e dare qualsiasi comando lavorando con il 64 come se nulla fosse (a parte un certo rallentamento...). Per rimettere tutto a posto dovrete premere Run - Stop / Restore.

Ovviamente ognuno può inserire in interrupt le routines che crede al posto degli esempi presentati, a patto che queste non superino la durata di 1/60 di secondo, e che abbiano, come ultima istruzione, un JMP $EA31 per consentire le normali funzioni di interrupt. Bisogna fare attenzione anche alle locazioni di memoria usate dalle varie routines.

Dal momento che queste lavorano praticamente in contemporanea, per non creare conflitti sarà utile far usare ad ogni routine le "sue" variabili e locazioni, diverse e separate da quelle di ogni altra. Questo per evitare che una routine modifichi locazioni usate, un attimo prima, da un'altra routine.

A questo punto credo si possa chiudere (per ora) l'argomento, rimandando il tutto alla lettura e comprensione dei vari listati e tabelle presenti in queste stesse pagine.

TUTTO IL MACROASSEMBLER CHE VOLETE

Una manuale di pronto intervento per chi smanetta in linguaggio macchina

di **Domenico Pavone**

EDITOR

Caricamento: Load "editor64", 8, 1
Attivazione: Sys 49152

COMANDI IMPARTIBILI
SOLO IN MODO DIRETTO

AUTO	Auto nr	Abilita la numerazione automatica delle linee, con nr che indica il valore di incremento. Per escluderla, impartire il comando senza alcun parametro.
CHANGE	Change/xxx/yyy/,r1-r2 Change/xxx/yyy/	Cerca la stringa xxx, e la sostituisce automaticamente con la stringa yyy. Nella prima forma della sua sintassi, il comando includera' nella sua ricerca le sole righe comprese tra i numeri r1 ed r2. Se, invece, l'ultimo parametro viene omesso, il 'replace' interessera' l'intero listato.
CPUT	Identica a Put	Salva il listato sorgente come Put, ma senza eventuali spazi superflui in esso contenuti. Il file occupera' un minor numero di blocchi sul disco, ma a scapito di una visione ordinata del disassemblato, se ricaricato con Get.
DELETE	Delete r1-r2 Delete r1- Delete -r2	Elimina dal listato il gruppo di righe specificato da r1 ed r2 (numeri di linea). I parametri riguardanti i numeri di riga, vanno inseriti secondo le ben note modalita' legate all'uso del comando basic List.
FIND	Find/xxx/,r1-r2 Find/xxx/	Cerca la stringa xxx e, per ogni ricorrenza trovata, ne visualizza l'intera linea. Se specificato (con r1-r2), la ricerca avviene nel range compreso tra i due numeri di riga, altrimenti viene esaminato l'intero listato.
	Format	Funziona esattamente come un List, solo che il testo viene visualizzato secondo la tipica suddivisione in campi dell'Assembler.

Pubblichiamo, in queste pagine, una tabella che verrà apprezzata soprattutto da chi, possedendo il manuale originale in inglese del Macroassembler Commodore, ha difficoltà ad utilizzare i vari comandi del potente assemblatore.

Come per tutte le tabelle che si rispettano, anche questa necessita di un po' di pazienza per esser digerita. Il "formato" è quello tradizionale: viene indicato dapprima il comando, poi qualche esempio di sintassi specifica, infine una esauriente spiegazione del funzionamento del comando stesso.

Vi consigliamo di fotocopiare queste pagine e di tenere il manualetto a portata di mano.

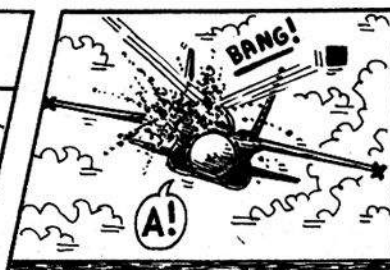
FORMAT	Format r1-r2	Lo scroll puo' essere interrotto e ripristinato premendo la barra spaziatrice, mentre i numeri di riga r1 ed r2 sono vincolati alla stessa sintassi di List.
GET	Get"nomefile" Get"nomefile",r	Carica nell'editor un file sorgente, precedentemente salvato con Put (o Cput). Nella sua forma piu' semplice (Get"file") il file sara' numerato a partire da 1000, con incremento 10, e provocando la cancellazione di un eventuale listato presente in memoria. Inserendo il parametro r (=numero di riga), la numerazione iniziera' a partire da questo valore, senza alterare eventuali altre righe gia' presenti in memoria, purche' di numerazione inferiore. In pratica, e' consentito un 'append' del file su disco al listato in elaborazione.
KILL	Kill	Esce dall'editor, che resta comunque a disposizione in memoria (purche' non vengano manipolate le locazioni da 49152 a 51199). Per riattivarlo, e' sufficiente impartire una nuova Sys 49152.
LIST	Come da Basic	Identico alla sua implementazione in Basic.
NUMBER	Number Number r1,r2,r3	Rinumera il listato in memoria. Senza alcun parametro, la numerazione iniziera' da 1000 con incremento 10. In caso contrario, r1 indica il numero della riga in memoria dal quale iniziare, r2 il nuovo numero che essa assumera', ed r3 l'incremento tra una riga e l'altra.
PUT	Put"nomefile" Put"nomefile",r1-r2	Salva il listato in memoria su disco, generando un file sequenziale (file oggetto). Specificando i numeri di riga r1 ed r2 (sempre secondo le stesse modalita' di List), puo' essere inviata al drive la sola sezione del sorgente compresa tra i due valori.

DIRETTIVE DI ASSEMBLAGGIO

USABILI SOLO NEL CONTESTO
DEL LISTATO ASSEMBLER

N.B. Tutte le istruzioni possono essere abbreviate nei loro primi 3 caratteri.

=	BASIC =$0801	Assegna alla label 'Basic' il valore $0801.
*=	*=$C000	Segnala all'assembler la prima locazione dalla quale iniziare l'assemblaggio. Nell'esempio, 49152.
=	*=*+10	Riserva un certo numero di byte in memoria. Nel punto in cui l'assembler incontrera' la direttiva mostrata nell'esempio, continuera' l'assemblaggio dieci byte piu' avanti.

.BYTE	.BYTE 5,$A,%1111 .BYTE 'stringa'	Indica all'assembler di inserire in memoria quanto specificato dall'operando, che puo' essere un numero (preceduto eventualmente dal simbolo indicante la sua notazione), o un carattere Ascii, indicato tra due apici. Nel primo esempio, vengono riservate 3 locazioni di memoria, inizializzate coi valori 5 (decimale), 10 ($A in esadecimale) e 15 (in binario). Nel secondo caso, verranno 'pokate' 7 locazioni (STRINGA e' composta da 7 caratteri) con i singoli codici Ascii di quanto racchiuso tra apici (tasto Shift+7).
.WORD	.WORD $FF01	Funziona piu' o meno come .Byte, ma riserva due locazioni di memoria per volta, e non sono ammessi caratteri Ascii. La caratteristica principale di questa direttiva e' quella di memorizzare l'operando in formato basso/alto, per cui, nell'esempio, verranno inizializzate due locazioni contenenti la prima il valore 1, e la seconda 255 (FF). Come per .Byte, possono essere inseriti piu' operandi dopo l'istruzione, purche' separati da una virgola, e in qualsiasi notazione.
.DBYTE	.DBYTE $FF01	Identico a .Word, con la sola differenza che i due byte vengono memorizzati nel loro formato reale, cioe' alto/basso. Nel punto in cui e' inserita questa istruzione, saranno dunque riservate due locazioni contenenti nell'ordine 255 (FF) ed 1.
.SKIP	.SKIP 2	Senza alcun parametro, produce una linea vuota nel listato generato dall'assembler, apprezzabile soprattutto quando si sceglie di avere una sua copia su carta. Nell'esempio, vengono generate due linee vuote, come specificato dal parametro che segue la direttiva.
.END	.END	Indica la fine del listato da assemblare, e deve essere presente come ultima istruzione del file sorgente.
.FIL	.FIL altrolistato	Conclude l'assemblaggio del file corrente, e aggancia di seguito a questo il codice prodotto dall'assemblaggio del file specificato come operando (nell'esempio, di nome 'altrolistato'). In pratica, consente un 'link' con altri files presenti sullo stesso dischetto, precedentemente elaborati con l'editor.
.LIB	.LIB altrolistato	Consente di inserire, nel punto in cui la direttiva e' invocata, l'assemblaggio di un file esterno, purche' anch'esso memorizzato sullo steso dischetto. A differenza di .Fil, dopo il 'link' del file esterno, l'assembler torna al listato principale, riprendendo l'elaborazione dal punto in cui l'aveva prima interrotta. Molto utile per gestire 'librerie' di routines personalizzate.

.MAC	.MAC nomedellamacro	Definisce una Macro Istruzione, ovvero un insieme di mnemonici il cui codice macchina puo' essere inserito in qualunque punto del listato sorgente, semplicemente citando il nome della Macro. All'interno di essa, possono essere passati dei parametri di volta in volta diversi, indicati da un punto interrogativo seguito da un numero (la posizione del parametro dopo la citazione della Macro (vedi CCC n.61)).
.MND	.MND	Deve concludere la serie di istruzioni che compongono una Macro (MND=Macro eND).

MONITOR

Caricamento : Load "Monitor$8000", 8, 1
Attivazione : Sys 32768

Caricamento : Load "Monitor$C000", 8, 1
Attivazione : Sys 49152
N.B.: Tutti i parametri numerici sono da intendersi in notazione esadecimale e vanno sempre espressi tramite 2 o 4 cifre (p.es. 08, 0435, ecc.)

A (assemble)	A C000 LDA #$00	Assembla all'indirizzo di memoria $C000 l'istruzione Assembly Lda (LoaD Accumulator) seguita dall'operando #$00. Dopo la pressione del Return, il monitor provvede automaticamente ad aggiornare l'indirizzo, che non va dunque reimmesso qualora si assemblino istruzioni successive.
C (compare)	C 0400 0800 7000	Confronta due aree di memoria: le locazioni da $400 a $800, con quelle da $7000 in poi (fino a $7400). Se risultano perfettamente identiche, riappare il solito punto che fa 'prompt'. In caso contrario, vengono visualizzati gli indirizzi dei byte che non coincidono.
D (disassemble)	D C000 D000	Legge il codice macchina (numerico) contenuto nelle locazioni di memoria da $C000 a $D000, e, se possibile, lo trasforma nei corrispondenti codici mnemonici ed operandi dell'Assembly. Si puo' anche disassemblare una singola istruzione (p.es. D C000), e poi visualizzare quelle seguenti (o precedenti) facendo scrollare lo schermo verso l'alto (o verso il basso) adoperando i tasti cursore.
F (fill)	F 2000 3000 EA	'Poka' il byte specificato dal terzo parametro (EA) in tutte le locazioni comprese tra $2000 e $3000 (Fill=riempi).
G (go)	G C000	Esegue la routine memorizzata apartire dall'indirizzo $C000. Per riprendere il controllo del sistema, e' opportuno inserire nella routine un break (brk) nel punto in cui si dsidera un ritorno [illegible]

		Comando molto delicato: salvare sempre la routine prima di eseguirlo.
H (hunt)	H C000 D000 A9 03 H C000 D000 'STRINGA	Effettua una ricerca nelle locazioni di memoria comprese tra $C000 e $D000. Nel primo esempio, verra' cercata la ricorrenza dei numeri esadecimali A9 e 3 (possono essere inclusi anche piu' di due byte, purche' separati da uno spazio). Adoperando invece il carattere di singolo apice, puo' essere edoperata come paragone della ricerca una stringa di caratteri. Se l'operazione ha esito positivo, il monitor visualizzera' gli indirizzi contenenti la chiave di ricerca.
I (interrogate)	I C000 D000	Effettua un 'dump' della memoria compresa nel range specificato dai due parametri. Sullo schermo, il contenuto di ogni cella viene mostrato tanto in formato esadecimale quanto in rappresentazione Ascii (in reverse). Facendo scrollare lo schermo tramite i tasti cursore, possono essere esaminate le locazioni che seguono o che precedono gli indirizzi inseriti nel comando.
L (load)	L "nomefile",08 L "nomefile",01	Carica in memoria un file, prelevandolo da disco (08) o nastro (01). Il file verra' memorizzato a partire dalla locazione specificata dall'header, ovvero i suoi due primi byte. In pratica, corrisponde al Load"file",8,1 del Basic.
M (memory)	M C000 D000	Stessa funzione del comando I, ma senza la traduzione Ascii dei singoli byte.
N (new locator)	N 2000 3000 1000	Modifica tutti gli indirizzi assoluti contenuti tra $2000 e $3000, aggiungendo loro il valore $1000 (offset). Dopo l'offset, possono essere aggiunti altri due parametri, che limitano la modifica agli indirizzi compresi nel range fornito con questi ultimi. Molto utile per rilocare routines (p.es.) trasferite con T, ma piuttosto complesso da usare (si veda n.66 della rivistá).
R (register)	R	Mostra il contenuto dei registri del 6502. Nell'ordine: Program Counter, Registro di stato, Accumulatore, Registro X, Registro Y, Stack Pointer.
S (save)	S "file",08,C000,D001 S "file",01,C000,D001	Salva su disco (08) o nastro (01) il contenuto della memoria compreso tra le locazioni $C000 e $D000, assegnando al file il nome incluso tra virgolette. Si badi che la locazione finale da salvare deve essere aumentata di una unita' nella specifica dell'ultimo parametro. Il file cosi' salvato, puo' essere ricaricato anche da basic con la forma Load...,8,1.
T	T C000 D000 8000	Trasferisce il contenuto dell'area di memoria compresa tra $C000 e $D000 nelle locazioni a partire da $8000. A differenza degli indirizzamenti relativi, quelli assoluti non vengono rilocati.

UN POINTER IN TECHNICOLOR

Avviciniamoci all'Assembly di Amiga; magari mediante un brevissimo programma

di **Donato de Luca**

Non abbiamo certo intezione di affrontare un corso sull'assembler del 68000, microprocessore dell'Amiga; per fare in modo che, tuttavia, tutti i nostri lettori possano capire il funzionamento del programmino proposto in queste pagine, dovremo spendere qualche parola per descrivere, sia pure sommariamente, il "cervello" dell'ottimo computer.

Il Motorola 68000 è un microprocessore a 16 /

32 bit. Possiede 8 registri destinati a contenere i dati (denominati D0. D1. D2, D3, D4, D5, D6, D7). 7 registri per gli indirizzi (A0, A1, A2, A3, A4, A5, A6) 2 puntatori allo stack (A7), un contatore di programma (PC) ed un registro di stato (SR).

Stavolta, per fortuna, gli unici registri che interessano sono quelli dei dati, degli indirizzi ed il registro di stato.

A PROPOSITO DI REGISTRI

Un registro è un *campo di memoria*, presente all'interno dello stesso 68000, in cui è possibile, ad alta velocità (rispetto ad una qualsiasi locazione Ram) immagazzinare un dato che può essere costituito da un solo byte, una word (= parola) o una "long word" il cui significato è spiegato altrove.

In un registro dati si può scrivere (e/o leggere) un qualsiasi dato. Anche in un registro indirizzi si può fare la stessa cosa. Tuttavia nel registro indirizzi il dato può essere considerato come un dato vero e proprio oppure come un *puntatore* ad una cella di memoria in cui rintracciare il dato effettivo. Prima di illustrare un esempio chiarificatore, è indispensabile parlare di una delle più note ed usate istruzioni dell'assembler del 68000, cioè *MOVE*.

La sua sintassi, indicata simbolicamente in quasi tutti i testi di informatica, è la seguente:

MOVE sorgente, destinazione

...che in parole povere significa:
"COPIA il valore dell'operando *sorgente* nell'operando *destinazione*".

Diciamo subito che questa è l'istruzione che userete più spesso se deciderete (come ce lo auguriamo...) di imparare a programmare in linguaggio macchina con l'Amiga.

Bene, ora passiamo all'esempio. Supponiamo che nel registro dati D1 sia presente il valore 2000. Con l'istruzione...

Move D1, 4000

...viene copiato il valore presente nel registro dati D1 (cioè 2000) nella locazione di memoria 4000.

Ora supponiamo che nel registro indirizzi A1 sia presente il valore 2000. Con l'istruzione...

Move A1, 4000

```
' Versione Basic

INIZIO:

        X=PEEKW(14675978)
        POKEW 14676386,X
        POKEW 14676388,X
        POKEW 14676390,X
        L=MOUSE(0)
        IF L=1 GOTO ESCI
        GOTO INIZIO

ESCI:
```

...otteniamo ancora lo stesso effetto di prima dal momento che, ora, nella locazione di memoria 4000 vi è il valore 2000. Supponiamo ancora che nel registro indirizzi A1 sia il valore 2000. L'istruzione...

Move (A1), 4000

...(notare le parentesi!) copia il valore presente nella locazione di memoria 2000 nella locazione di memoria 4000. Quindi abbiamo copiato il valore della locazione di memoria a cui stava *puntando* il registro indirizzi A1 (in questo caso alla locazione 2000) nella locazione di memoria 4000. Se, ad esempio, nella locazione di memoria 2000 era presente il valore 15, dopo l'esecuzione dell'istruzione stessa il valore 15 sarà presente nella locazione di memoria 4000. Per utilizzare un registro indirizzi come puntatore a una locazione di memoria, quindi, lo si deve porre tra due parentesi tonde.

E' bene precisare che i valori riportati nell'esempio appena descritto sono puramente indicativi ed è quindi possibile, alterando a casaccio il valore presente nella locazione 2000 (o anche nella 4000) cadere nella trappola mortale di un Guru. Questo può accadere in quanto è possibile che, in una di quelle locazioni, sia presente la parte vitale di un programma.

Come avremo modo di vedere nei prossimi numeri di C.C.C. le "zone" di memoria dell'Amiga non

L'Assembly di Amiga è davvero complesso; studiamolo un po' per volta

PER QUALCHE PAROLA IN PIU'

Il micro 68000 conosce quattro tipi di dati:

BIT n
...in cui *n* è il numero del bit. Un dato di questo tipo può assumere 2 exp 1 valori, cioè può assumere i soli valori zero (0) ed uno (1).

BYTE (.b) 8 BIT (0 - 7)
...che può assumere 2 exp 8 valori, cioè valori compresi tra 0 e 255.

WORD (.w) 16 BIT (0 - 15)
...che può assumere 2 exp 16 valori, cioè può assumere valori compresi tra 0 e 65535

LONGWORD (.l) 32 BIT (0 - 31)
...che può assumere 2 exp 32 valori, cioè può assumere valori compresi tra 0 e 4294967299 (e se riuscite a leggerlo senza impaperarvi siete bravi).

Di solito, se si lavora solo con i registri, la dimensione dei dati non ha molta importanza; tuttavia si deve tener conto che:

- Non è possibile copiare un singolo bit in un registro.
- Non è possibile copiare un byte in un registro indirizzi.

Inoltre, nel programmino presentato, è importante usare sempre il formato word. Infatti i registri hardware dei colori sono registri a 16 bit (1 word)

La routine l.m. di queste pagine è in "formato" Seka

sono predeterminabili come nel caso del C/64. Tutto si muove sotto... le vesti di Amiga: un dato che, in un certo istante, è presente in una certa locazione di memoria, potrebbe non esser più presente nell'istante immediatamente successivo. E' difficile crederlo, ma è proprio così!

Il 68000 offre numerosi tipi d'indirizzamento, uno più potente dell'altro. Per capire il funzionamento del programmino proposto in queste pagine, tuttavia, ci limiteremo a studiarne uno solo, quello *diretto*.

Come al solito ricorriamo ad un semplice esempio. Supponiamo che nel registro dati D1 sia contenuto il valore 5, mentre nel registro dati D0 sia contenuto il valore 6. L'istruzione...

Move D1, D0

...copia il valore contenuto nel registro dati D1 nel registro dati D0. Se, quindi, prima di eseguire l'istruzione era presente il valore 5 in D1 ed il valore 6 in D0, dopo la sua esecuzione avremo 5 sia in D1 sia in D0.

IL PROGRAMMA ASSEMBLY

E, finalmente, esaminiamo il programma proposto, senza vergognarci di precisare anche le questioni più ovvie e banali.

La prima linea che leggiamo è denominata *Inizio* (per la versione SEKA Assembly). Questo termine non è un'istruzione del microprocessore 68000, ma è una *Label*. Una label ("etichetta", in inglese) è un qualsiasi nome di fantasia ("Inizio", nel nostro caso) che viene assegnato (dallo stesso assemblatore) ad un indirizzo di memoria in modo da rintracciarlo, in seguito, con la massima facilità.

Quindi la label indica (almeno nel nostro caso

particolare) il "luogo" in cui è localizzato il programma nella memoria del computer.

La seconda linea che leggiamo (Move.W $DFF00A, D0) è, invece, una vera e propria istruzione. La particolare sintassi di tale istruzione (.W) si riferisce all'intera word, presente nel registro hardware $DFF00A, che viene ricopiata nel registro D0 (vedi riquadro specifico).

Se avessimo usato, al posto dell'istruzione *Move.W*, l'istruzione *Move.B* avremmo copiato solo il primo byte della word contenuta nel registro hardware $DFF00A; se avessimo, ancora, usato l'istruzione *Move.L* avremmo copiato, nel registro D0, sia il contenuto del registro hardware $DFF00A sia di quello successivo $DFF00C.

Ciò sarebbe avvenuto in quanto i registri hardware $DFF00A e $DFF00C sono posti a 16 bit di distanza l'uno dall'altro ed è ovvio che, copiando 32 bit partendo dal primo registro, verrebbe copiato anche il valore presente nel secondo registro.

Il registro hardware $DFF00A corrisponde alla porta Joystick n. 0, cioè quella del mouse. Ciò vuol dire che, dopo aver "lanciato" il programma, tutte le volte che muoveremo il mouse il valore contenuto in questo registro cambierà, dal momento che "segue" le sorti del mouse stesso. La trascrizione del valore contenuto nel registro hardware $DFF00A, nel registro dati D0, viene effettuata per registrare, il valore stesso, in un posto sicuro. Infatti il registro hardware $DFF00A fa parte di quel gruppo di registri i quali, una volta letti, vengono sempre azzerati; subito dopo, se non bastasse, vengono sovrascritti nuovi valori: non sono certo registri degni di contenere informazioni riservate e preziose!

Continuiamo ad esaminare il nostro programma. L'istruzione...

Move.W D0, $DFF1A2

...copia la word contenuta nel registro dati D0 (cioè il valore che, precedentemente, era contenuto nel registro hardware $DFF00A) nel registro hardware $DFF1A2. L'istruzione...

Move.W D0, $DFF1A4

...svolge, in pratica, la stessa operazione, solo che D0 viene copiato in $DFF1A4. L'istruzione...

Move.W D0, $DFF1A6

```
Versione Assembler (sintassi Seka)
INIZIO
        MOVE.W $DFF00A,D0
        MOVE.W D0,$DFF1A2
        MOVE.W D0,$DFF1A4
        MOVE.W D0,$DFF1A6
        BTST #6,$BFE001
        BEQ ESCI
        JMP INIZIO
ESCI:
        RTS
```

...svolge anch'essa la stessa operazione, solo che D0 viene copiato in $DFF1A6.

A che servono queste istruzioni?

Anche per gli irriducibili del Basic è a disposizione una routine che li soddisfa.

IL MISTERO DEI 3 REGISTRI

Vediamo di ricapitolare. A questo punto delle operazioni il registro $DFF1A2 contiene il valore del colore 17; il registro $DFF1A4 contiene il valore del colore 18 e $DFF1A6 contiene il valore del colore 19.

Chi conosce, anche superficialmente, la gestione grafica di Amiga, avrà capito perchè vengono cambiati i valori presenti nei tre registri hardaware $DFF1A2, $DFF1A4, $DFF1A6.

Chi, invece, non ne sa proprio niente (non possiamo certo pretendere che tutti siano perfetti come noi..) farà bene a leggere con la massima attenzione le note che seguono.

Gli amighi conoscono di certo il *pointer* ("puntatore", la freccetta insomma) del mouse. Questo non è altro che un pacifico sprite, ed esattamente quello numerato con zero.

Nell'Amiga si hanno a disposizione ben quattro coppie di due sprite hardware ciascuna (in questo caso il termine "hardware" è usato propriamente), cioè la coppia 0 - 1, la coppia 2 - 3, la coppia 4 - 5 e la coppia 6 - 7.

La coppia che interessa particolarmente è la prima, quella che comprende lo sprite 0. Per ogni cop-

DAL SOFTWARE ALL'HARDWARE

Nel corso dell'articolo i registri $DFF00A, $DFF1A2, $DFF1A4, $DFF1A6, $BFE001 sono stati definiti registri di tipo *hardware*.

Alcuni lettori si saranno certamente chiesti la differenza esistente tra questi registri ed i registri del 68000 (cioè D0, D1, D2, D3, D4, D5, D6, D7, A0, A1, A2, A3, A4 A5, A6, A7 O SP, PC, SR).

La differenza consiste nel fatto che i registri D0 - D7 / A0 - A7, PC, SR sono contenuti all'interno del 68000, mentre i registri $DFF00A, $DFF1A2, $DFF1A4, $DFF1A6 sono contenuti in Denise, coprocessore che affianca il 68000. Il registro $BFE001 è contenuto nell'8520-A, uno dei due CIA (Complex Interface Adaptator).

Vengono riportati, in queste stesse pagine, anche i 31 registri colore di Amiga.

Mappa memoria dei 31 registri colore di Amiga

Nome registro	Indirizzo	Commento
COLOR 00	$DFF180	Colore di fondo
COLOR 01	$DFF182	
COLOR 02	$DFF184	
COLOR 03	$DFF186	
COLOR 04	$DFF188	
COLOR 05	$DFF18A	
COLOR 06	$DFF18C	
COLOR 07	$DFF18E	
COLOR 08	$DFF190	
COLOR 09	$DFF192	
COLOR 10	$DFF194	
COLOR 11	$DFF196	
COLOR 12	$DFF198	
COLOR 13	$DFF19A	
COLOR 14	$DFF19C	
COLOR 15	$DFF19E	
COLOR 16	$DFF1A0	Colore 00 (trasparente)
COLOR 17	$DFF1A2	Colore 01 (Prima coppia sprites)
COLOR 18	$DFF1A4	Colore 02 (" ")
COLOR 19	$DFF1A6	Colore 03 (" ")
COLOR 20	$DFF1A8	Colore 00 (trasparente)
COLOR 21	$DFF1AA	Colore 01 (Seconda coppia sprites)
COLOR 22	$DFF1AC	Colore 02 (" ")
COLOR 23	$DFF1AE	Colore 03 (" ")
COLOR 24	$DFF1B0	Colore 00 (trasparente)
COLOR 25	$DFF1B2	Colore 01 (Terza coppia sprites)
COLOR 26	$DFF1B4	Colore 02 (" ")
COLOR 27	$DFF1B6	Colore 03 (" ")
COLOR 28	$DFF1B8	Colore 00 (trasparente)
COLOR 29	$DFF1BA	Colore 01 (Quarta coppia sprites)
COLOR 30	$DFF1BC	Colore 02 (" ")
COLOR 31	$DFF1BE	Colore 03 (" ")

pia di sprite vi sono quattro registri relativi agli altrettanti colori della stessa coppia di sprite, che, in realtà, sono tre in quanto il primo colore di ogni coppia è il colore *trasparente* della coppia. Per motivi tecnici, infatti, i colori dello sprite 0 sono uguali a quelli dello sprite 1, così come quelli dello sprite 2 sono uguali a quelli dello sprite 3: la stessa cosa vale per le altre due coppie. Quindi i registri hardware $DFF1A2, $DFF1A4, $DFF1A6 contengono i colori dello sprite 0; e quindi del pointer.

E' ora facile(!) indovinare la funzione del programmino di queste pagine: tutte le volte che il mouse verrà spostato, il pointer (sprite 0) muterà il proprio colore.

IL REGISTRO DI STATO

In precedenza, descrivendo i registri del 68000, abbiamo accennato al registro ST, lo Status Register.

Questo è un registro a 16 bit i cui primi cinque sono i flag di stato. Un flag è, praticamente, un bit e può quindi assumere due valori: zero oppure uno.

Eseguendo alcune istruzioni, il micro 68000 modifica uno, o più, flag di stato a seconda del tipo d'istruzione e/o del risultato ottenuto. Il flag di stato che ci interessa da vicino è quello denominato "Z", vale a dire il bit 2 del registro di stato SR; e vedremo subito il perchè.

L'istruzione successiva (stiamo ancora commentando il programmino l.m.) è...

BTST #6, $BFE001

...che controlla il sesto bit del registro hardware $BFE001 (il perchè lo vedremo dopo). Nel caso in cui il bit 6 valga 1, pone a 0 il flag Z; se, invece, vale 0, pone a 1 il flag Z.

L'istruzione successiva (BEQ ESCI:) è una di quelle istruzioni dette di diramazione, che, cioè, obbligano ad effettuare un "salto" all'indirizzo specificato (nel nostro caso, quello assegnato dall'assemblatore alla label ESCI:) se la condizione specificata è soddisfatta. La condizione specificata è la condizione Branch Equal (EQ), che sarà verificata quando il flag Z varrà 1. Se, quindi, il flag Z vale uno (e, di conseguenza, il bit 6 del registro hardware $BFE001 vale 0), il programma salterà alla label Esci. Se, invece, il flag Z vale 0, il programma continua con la successiva istruzione, cioè...

JMP INIZIO:

...la quale impone un ritorno all'inizio dello stesso programma.

Il bit n. 6 del registro hardware $BFE001 vale 1 se il pulsante sinistro del mouse non è premuto; se, invece, è premuto, esso vale 0.

Quindi, controllando questo bit, possiamo sapere quando il pulsante sinistro del mouse è premuto; in tal caso possiamo far saltare il programma ad una determinata label, (ESCI: nel nostro caso) in cui incontra l'istruzione RTS che, in pratica, restituisce il controllo al computer.

IN BASIC

Il programma prima descritto è riportato nella versione *Seka Assebler*; tuttavia ci rendiamo conto che non tutti il lettori possiedono un Assemblatore e pubblichiamo la corrispondente "traduzione" in AmigaBasic.

Dal momento che la notazione usata in Assembler è quella decimale (individuabile dal simbolo del dollaro, $), nella versione Basic del programma i valori dei registri hardware saranno espessi in notazione decimale. La traduzione è semplice da effettuare, ma preferiamo risparmiarvi la fatica:

$DFF00A = 14675978
$DFF1A2 = 14676386
$DFF1A4 = 14676388
$DFF1A6 = 14676390

Il registro hardware $BFE001, nella versione Basic, non viene testato direttamente, come invece avviene con il registro $DFF00A.

Esaminiamo ora, rapidamente, la versione Basic. La label INIZIO indica (guarda caso) l'inizio del programma. Il comando...

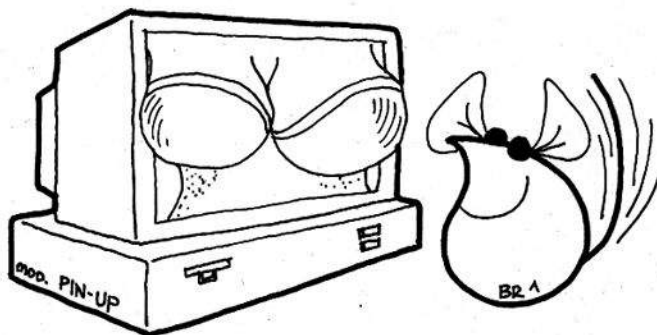
X = PEEKW (14675978)

...copia il valore presente in 14675978 (ex $DFF00A) in X. Da notare che la sintassi PEEKW consente di copiare l'intera word.

Le tre *PokeW* successive copiano il valore di X nei registri hardware $DFF1A2, $DFF1A4, $DFF1A6. L'istruzione L = mouse(0) controlla se il tasto sinistro del mouse è stato premuto e se ciò avviene (if L=1) si salterà (Goto ESCI) alla fine. In caso contrario (Goto INIZIO) il programma riprenderà dall'inizio.

UN RIEMPIMENTO PERSONALIZZATO

Il comando Paint, a disposizione nel C/128, colora in modo uniforme intere aree hi - res; la nostra routine, invece...

di **Armando Sforzi**

Una routine velocissima personalizza le schermate hi-res del vostro C/128

La routine di queste pagine è una variante dell'istruzione Paint presente nel Basic 7.0 del C/128 e possiede un'interessante quanto utilissima caratteristica: permette di riempire un'area specificata con il disegno di una maschera altettanto prestabilita.

Questa funzione, presente nei linguaggi e nei sistemi più evoluti, non figura purtroppo sui "mini" Commodore, il cui comando Paint, là dove esiste, si limita a riempire con piatta uniformità una determinata superficie, rendendo impossibile l'individuazione dei limiti di aree confinanti, necessità che, invece, emerge frequentemente quando si ha a che fare con grafici, specialmente istogrammi ed areogrammi, o disegni di vario genere.

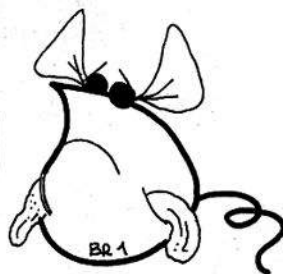
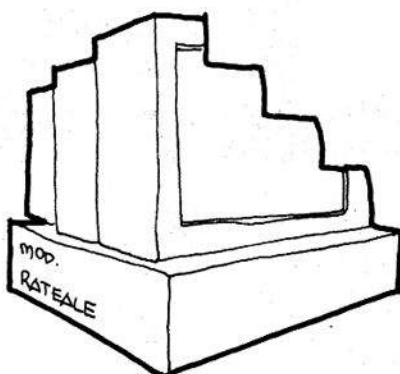
Con la routine di queste pagine si supplisce brillantemente al problema facendo in modo che ciascuna regione assuma un "lay-out" diverso e ciò, oltre a favorire la praticità della lettura, conferisce al disegno un aspetto estetico decisamente professionale.

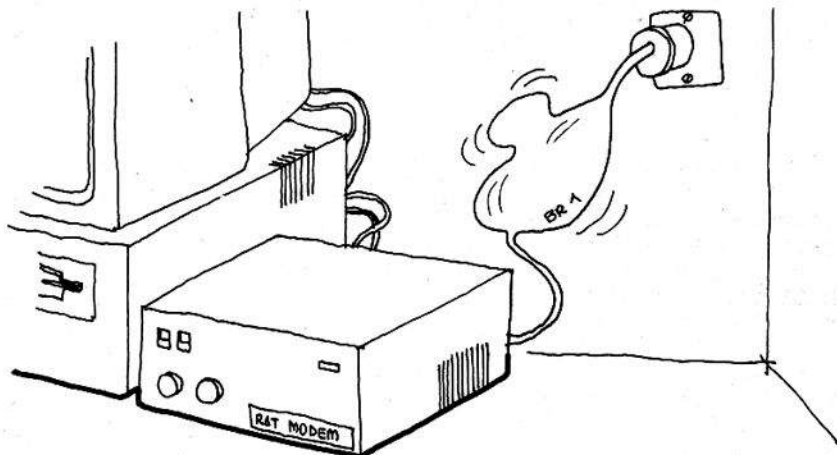
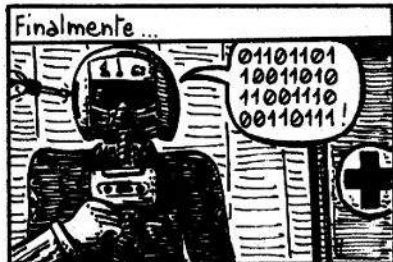
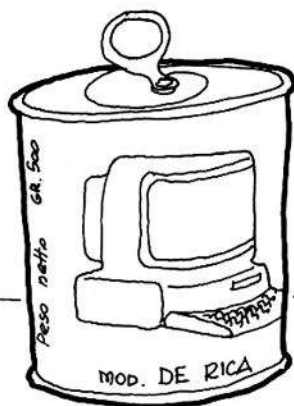
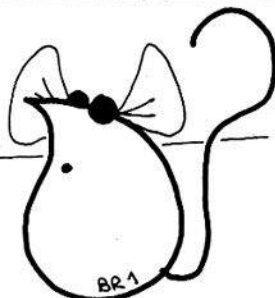
La struttura della routine è descritta nel disassemblato commentato, mentre la sintassi operativa è...

SYS 4864: x, y, indmaschera

...in cui:

X, Y sono le coordinate di un punto, interno alla figura, da cui si inizia il riempimento (vedi riferimento all' istruzione Paint del Basic);
..indmaschera è l'indirizzo del primo di una sequenza di otto bytes, posti in pagina zero, che contengono la maschera di riempimento.

Viene pubblicato anche un demo ed un piccolo programma per preparare maschere che soddisfino le proprie necessità e i propri gusti.

```
0 PRINT"  PAINT. SOLO PER C/128
1 PRINT"  BY   ARMANDO SFORZI
2 PRINT"  BOZZANO  (LU)  1989
9 :
10 PRINT" SYS4864,X,Y,INDMASCHERA
20 PRINT" X,Y: COORDINATE DI UN PUNTO INTERNO ALLA FIGURA
30 PRINT" INDMASCHERA: INDIRIZZO IN PAGINA 0 DEL LAY-OUT
89 :
90 S=0:FORJ=0TO146:READA:POKE4864+J,A:S=S+A:NEXT
95 IFS<>15351THENPRINT"ERRORE NEI DATA"
98 END
99 :
100 DATA032,123,019,140,049,017,141,050,017,032,123,019,140,051,017
110 DATA141,052,017,032,123,019,132,251,133,252,032,130,019,134,039
120 DATA177,038,145,112,200,208,249,230,113,165,113,201,254,208,003
130 DATA076,058,077,232,224,064,208,231,032,208,097,032,130,019,134
140 DATA039,160,000,169,112,032,159,003,133,036,032,192,003,069,036
150 DATA072,049,251,133,037,104,073,255,037,036,005,037,145,038,200
160 DATA192,008,144,225,152,024,101,038,133,038,165,039,105,000,201
170 DATA064,240,014,133,039,152,024,101,112,133,112,144,199,230,113
180 DATA208,195,096,032,128,003,032,192,109,096,172,016,018,173,017
190 DATA018,132,112,133,113,160,000,162,032,132,038,096
200 END
```

```
C/128: Disassemblato commentato della routine Pattern Paint.
      Indirizzo di inizio: 4864  (1300)
. 01300  20 7B 13 JSR $137B       ; Carica dal comando...
. 01303  8C 31 11 STY $1131       ; i valori...
. 01306  8D 32 11 STA $1132       ;
. 01309  20 7B 13 JSR $137B       ;
. 0130C  8C 33 11 STY $1133       ;
. 0130F  8D 34 11 STA $1134       ;
. 01312  20 7B 13 JSR $137B       ;
. 01315  84 FB    STY $FB         ;
. 01317  85 FC    STA $FC         ; ...dei parametri
. 01319  20 82 13 JSR $1382       ; Salva la Hi/Res nel buffer
. 0131C  86 27    STX $27         ;
. 0131E  B1 26    LDA ($26),Y     ;
. 01320  91 70    STA ($70),Y     ;
. 01322  C8       INY             ;
. 01323  D0 F9    BNE $131E       ;
. 01325  E6 71    INC $71         ;
. 01327  A5 71    LDA $71         ; Controlla se c'e' memoria disponibile
. 01329  C9 FE    CMP #$FE        ;
. 0132B  D0 03    BNE $1330       ;
. 0132D  4C 3A 4D JMP $4D3A       ; Se non c'e' e' "OUT OF MEMORY"
. 01330  E8       INX             ;
. 01331  E0 40    CPX #$40        ;
. 01333  D0 E7    BNE $131C       ; Continua il ciclo di salvataggio
. 01335  20 D0 61 JSR $61D0       ; Lancia il comando BASIC di PAINT
. 01338  20 82 13 JSR $1382       ; ** Inizia la routine di Pattern Fill **
. 0133B  86 27    STX $27         ;
. 0133D  A0 00    LDY #$00        ;
. 0133F  A9 70    LDA #$70        ; Confronta i bytes della Hi/Res...
. 01341  20 9F 03 JSR $039F       ;
. 01344  85 24    STA $24         ; con quelli del buffer...
. 01346  20 C0 03 JSR $03C0       ;
. 01349  45 24    EOR $24         ; per rivelare quelli modificati...
. 0134B  48       PHA             ;
. 0134C  31 FB    AND ($FB),Y     ; dopo l' istruzione PAINT...
. 0134E  85 25    STA $25         ;
. 01350  68       PLA             ; ed assegna ad essi...
. 01351  49 FF    EOR #$FF        ;
. 01353  25 24    AND $24         ; il valore prelevato...
. 01355  05 25    ORA $25         ;
. 01357  91 26    STA ($26),Y     ; dalla maschera
. 01359  C8       INY             ; Continua il ciclo...
```

```
. 0135A   C0 08      CPY #$08     ; perche' la maschera...
. 0135C   90 E1      BCC $133F    ; e' lunga 8 bytes
. 0135E   98         TYA          ; Aggiorna i contatori...
. 0135F   18         CLC          ; per consentire il controllo...
. 01360   65 26      ADC $26      ; di tutta l' area Hi/Res
. 01362   85 26      STA $26      ;
. 01364   A5 27      LDA $27      ;
. 01366   69 00      ADC #$00     ;
. 01368   C9 40      CMP #$40     ;
. 0136A   F0 0E      BEQ $137A    ;
. 0136C   85 27      STA $27      ;
. 0136E   98         TYA          ;
. 0136F   18         CLC          ;
. 01370   65 70      ADC $70      ;
. 01372   85 70      STA $70      ;
. 01374   90 C7      BCC $133D    ;
. 01376   E6 71      INC $71      ;
. 01378   D0 C3      BNE $133D    ;
. 0137A   60         RTS          ; Esce dalla routine.
. 0137B   20 80 03   JSR $0380    ; ** Subroutine di prelievo... **
. 0137E   20 C0 6D   JSR $6DC0    ; dei parametri dal comando
. 01381   60         RTS          ;
. 01382   AC 10 12   LDY $1210    ; ** Subroutine di caricamento...**
. 01385   AD 11 12   LDA $1211    ; dei bytes contatori.
. 01388   84 70      STY $70      ;
. 0138A   85 71      STA $71      ;
. 0138C   A0 00      LDY #$00     ;
. 0138E   A2 20      LDX #$20     ;
. 01390   84 26      STY $26      ;
. 01392   60         RTS          ;
```

```
0 REM DEMO/PAINT/128
2 :
5 INPUT"NOME FILE PATTERN";X$
10 BLOAD (X$),B0,P5200:REM IND.ESEMPLIF.
20 COLOR0,2:GRAPHIC2,1
30 X=10:FORJ=0TO6:C=J+1:IFC=2THENC=14
40 COLOR1,C:BOX,X,10,X+35,40
50 SYS4864:X+20,20,5200+J*8
60 X=X+40:NEXT
70 END
```



```
1 REM PATTERN EDITOR
2 :
5 TRAP200
10 PRINTCHR$(147)"          *** PATTERN EDITOR ***"
12 PRINT:PRINT". PER DISEGNARE USA QUALSIASI CARATTERE"
13 PRINT". PREMI RETURN PER FINIRE"
20 CHAR,15,6,"........":FORJ=7TO14:CHAR,14,J,".        .":NEXT:CHAR,15,15,".....
..."
30 WINDOW15,7,22,14:OPEN1,0:POKE248,255
40 INPUT#1,I$:CLOSE1:PRINTCHR$(19)CHR$(19)
50 CHAR,0,17,"":PRINT". INDIRIZZO DI BUFFER ";:IFIB+C<>0THENPRINTIB+C
55 CHAR,21,17,"":INPUTIB:C=0:IFIB=0THEN50
60 FORJ=0TO7:S=0:FORK=0TO7:IFPEEK(1319+J*40+K)<>32THENS=S+2↑(7-K)
70 NEXT:POKEIB+C,S:C=C+1:NEXT
75 PRINTCHR$(18)"AREA SALVATA DA"IB" A"IB+C-1
80 CHAR,0,19,"":PRINT"1. EDITA":PRINT"2. SALVA":PRINT"3. FINE"
90 INPUT"1/2/3";J:ONJGOTO10,100
99 POKE248,0:END
100 INPUT"IND/INIZIO, IND/FINE";J,K:PRINTCHR$(145)CHR$(27)"Q";
110 INPUT"NOME FILE";I$:BSAVE(I$),B0,P(J)TOP(K+1):PRINTDS$:GOTO80
200 RESUMENEXT
```

Certamente, grazie alla perizia del dottore, Primo è SAVE... Resta però un problema: come mai in un'area di memoria da tutti ritenuta vuota era invece allocato il videogame "PONG", rivelatosi fatale nella missione di Primo Giovedini?

FINE DEL PRIMO FILE.

Vi ricordo che il secondo file di questa avventura sarà...

DAVIDE CONTRO GOLIA

Poche righe di Basic possono dare del filo da torcere anche agli sprotettori attrezzati con le famigerate cartucce sprotettrici

di **Lorenzo Emilitri**

Da qualche anno a questa parte, sono presenti sul mercato informatico decine e decine di dispositivi che promettono all'acquirente la possibilità di realizzare "copie totali" con metodi "imbattibili" per, naturalmente, "evitare che i propri originali si rovinino" (il fatto che tali sistemi vengano regolarmente utilizzati dai "ladri da edicola" per realizzare migliaia di copie di un programma di successo non sembra toccare minimamente l'animo candido ed innocente dei creatori di tali malefici dispositivi).

I primi esemplari erano piuttosto rozzi: salvavano su disco tutto il contenuto della memoria (suddiviso in vari spezzoni) ma avevano percentuali di riuscita ancora piuttosto basse. Successivamente sono comparse le cartucce della seconda generazione: più veloci delle precedenti, assicuravano percentuali di successo molto maggiori, salvavano in un solo file (spesso compattato, anche se troppo lungo per essere confuso con il lavoro di un vero sprotettore umano) ed includevano spesso vari toolkits che aiutano (di molto) sia il lavoro dei programmatori che quello degli hackers. In questi ultimi tempi stiamo assistendo alla nascita delle cartucce della terza generazione: veloci, potenti, dotate di toolkits super sofisticati: in una parola, imbattibili. Ma...

INTERROGHIAMO L'ORACOLO

Anche noi della mitica Redazione, stupefatti dal continuo proliferare di tali aggeggi (e forse ispirati da Sua Santità il nostro megadirettore genetrale), ci siamo chiesti: ma non c'è alcuna via di scampo da questa invasione degna dei Visitors?
Proviamo a pensare a cosa fa una cartuccia quando viene attivata: trasferisce su supporto magnetico tutta la memoria del 64 (o 128), ovvero 64k di RAM normale e 1k di memoria video, più un programmino di pochi bytes che si occupa del restart del programma copiato, generalmente utiliz-

zando in maniera appropriata lo stack.

"Ebbene, -starete pensando- visto che il programma deve risiedere da qualche parte in memoria, come può non essere copiato?"

Beh, è vero, il programma si deve trovare da qualche parte in memoria, e verrà pure copiato dalla cartuccia che effettua il tentativo, ma ciò non significa che esso debba per forza funzionare...

Anche se quanto detto sopra può sembrare piuttosto sibillino, non scoraggiatevi, tra poche righe tutto vi sarà chiaro.

DOCTOR 1541 E MISTER BASIC

Tutti i nostri lettori dovrebbero ormai sapere come è fatto un Floppy Disk Driver 1541: esso è un'unità "intelligente" ed è quindi in grado di eseguire autonomamente i comandi che gli vengono impartiti dal nostro C/64. Se è in grado di fare ciò, significa che dispone di un microprocessore (nel nostro caso un 6502), e se dispone di un microprocessore deve disporre di una zona dove il micro possa immagazzinare i suoi dati, appunto, una MEMORIA. E noi non stavamo forse cercando una memoria accessibile alla CPU ma che, contemporaneamente, non venisse copiata dalle cartucce sprotettrici? Elementare, mio caro Watson!

Ed il bello è che questa memoria è accessibile addirittura più facilmente da Basic che da Linguaggio Macchina, tanto che il programma presentato, perfettamente funzionante, è stato scritto completamente in Basic.

LEGGERE E SCRIVERE

Come dicevamo poco sopra, è perfettamente possibile leggere e scrivere direttamente nella RAM del drive anche da Basic.

Per fare ciò, è sufficiente inviare alcuni comandi al drive (utilizzando, naturalmente, il canale 15), tra cui:

M-W addr num dati

Permette di scrivere un valore (o più di uno) nella RAM del 1541; *addr* è l'indirizzo iniziale e va espresso nella consueta forma byte basso - byte alto; *num* è il numero di dati da trasferire e *dati* sono i dati stessi (da cosa l'avevate intuito?).

Naturalmente, tutti i valori devono essere passati al drive sotto forma di stringa, e quindi s'impone l'uso della funzione CHR$ (vedi listato pubblicato). Tra gli addetti ai lavori esiste una specie di convenzione su come inviare i dati: di solito (come nel nostro programma) si usa tenere fisso il valore di num ad 1 ed inviarli quindi singolarmente. Nulla però vieta di rivoluzionare l'uso passando i dati a due a due, fermo restando però che la lunghezza massima della stringa dei dati è di 34 caratteri.

M-R addr

Questo comando è invece il comando opposto, e permette di leggere (tamite una serie di GET# sul canale 15) il valore dei dati immagazzinati a partire dalla posizione addr.

LA VENDETTA DEL BASIC

Dopo le precedenti, noiose ma indispensabili note teoriche, possiamo finalmente dare un'occhiata più approfondita alla tecnica di protezione. Essa si basa sul fatto che, finchè il programma gira sul computer su cui è stato lanciato, la RAM del drive (o almeno la parte che interessa) deve contenere per forza i valori impostati.

Quando, poi, il programma viene copiato e inizia a girare su un altro computer, magari senza nemmeno un drive disponibile, il programma stesso si renderà conto di essere stato copiato, e qui iniziano le "contromisure", "ritorsioni" o "carognate" che dir si voglia. Il primo impulso in un caso del genere è sempre quello di inserire una routine di Fast Format in grado di cancellare irrimediabilmente il dischetto presente nel drive. Ma siete sicuri che sia questo il modo di apportare il massimo danno all'utilizzatore abusivo?

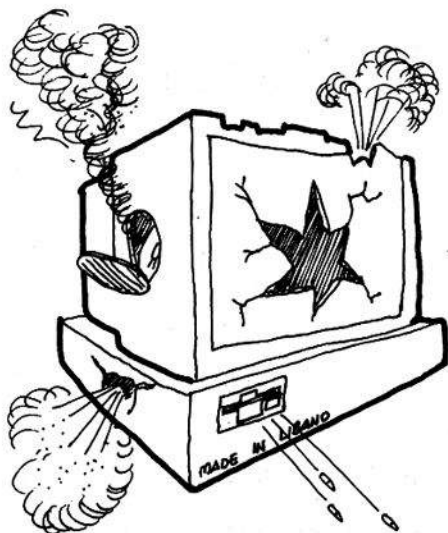
Prendiamo, ad esempio, un programma per l'amministrazione di un condominio, che dopo faticose notti sulla tastiera avete venduto al Signor A. Questo individuo, possedendo una cartuccia sprotettrice, ne ha venduto sottobanco una copia al Signor B, anche lui amministratore. Se questa copia gli cancellerà il disco su cui essa stessa è presente, egli potrebbe farsene dare un'altra e questa volta proteggere il disco dalla scrittura. O se il programma semplicemente si impianta, egli non utilizzerà il vostro software per il suo condominio ma rimarrà impunito lo stesso.

Ammettiamo per un momento che il vostro software, pur *sembrando* funzionare regolarmente, inizi a sbagliare i calcoli con precisione matematica: ad esempio, sbagli la spartizione di alcune spese o "dimentichi" alcuni dati essenziali: torme di inquilini inferociti gli insegneranno una volta per sempre a stare alla larga del software pirateggiato.

IL PROGRAMMA

Proprio questa è la tecnica che abbiamo scelto per il programma qui presentato: esso si occupa di effettuare delle moltiplicazioni tra due dati forniti dall'utente. Tutto funziona regolarmente finchè il programma non viene portato su un'altra macchina tramite una cartuccia sprotettrice: in questo caso esso inizierà a sbagliare i calcoli rendendolo, di fatto, inutilizzabile (riga 370).

Come si può vedere, per proteggere è sufficiente per prima cosa attivare la routi-

ne di scrittura e poi, periodicamente, saltare alla routine di lettura (ad esempio, in un gioco, alla fine della partita). Questa restituisce in CP il valore 0 se tutto è a posto, ed il valore 1 nel malaugurato caso che si tratti di una copia.

Ora, se il programma è strutturato come quello di queste pagine (per ovvie ragioni didattiche), è piuttosto facile, per un hacker, premere Run / Stop e farlo ripartire, ingannando così la routine di lettura che crederà che tutto sia a posto. Ma se il programma è stato compilato e non attiva la routine di scrittura se non con un salto ad una linea particolare (molti compilatori stampano l'indirizzo d'inizio del codice corrispondente alle varie linee Basic) le cose si complicano notevolmente per l'odioso individuo che sta cercando di derubarci del nostro lavoro.

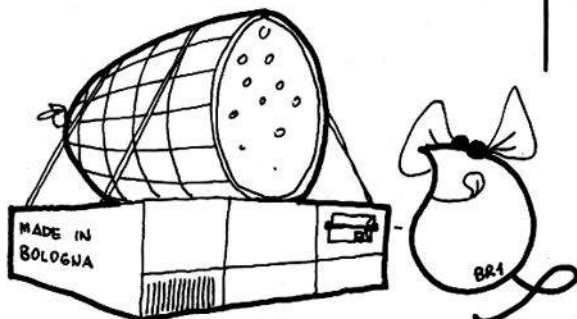
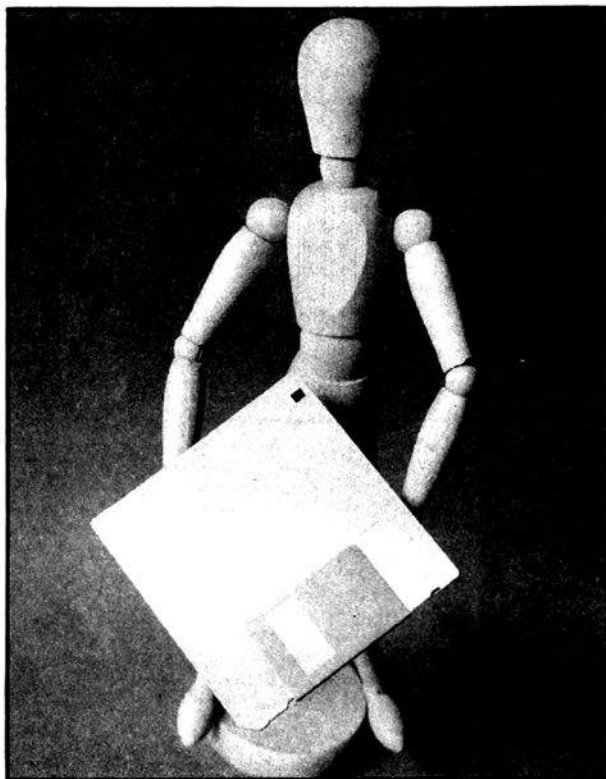
EFFETTI COLLATERALI

Questo programma funziona con la maggior parte delle cartucce oggi in commercio, anche se quello delle cartucce sprotettrici è un campo in continua evoluzione e quindi non possiamo garantirvi che funzioni con tutte. Però, vista la semplicità della tecnica e l'ampio spettro d'azione, siamo sicuri che valga la pena tentare.

La stringa è allocata nella memoria del drive a partire dalla locazione 193, ovvero un'area utilizzata per i files relativi. Quindi, se dovete utilizzare dei files relativi, è meglio per voi cambiare il valore della variabile MW nella linea 520 (dite la verità, quante volte avete utilizzato dei files relativi in tutta la vostra vita?).

Più probabile l'esistenza di incompatibilità con Fast Loaders, sistemi operativi paralleli e diavolerie simili: anche in questo caso consigliamo di cambiare il valore della variabile MW o, al limite, di rinunciare all'uso di questi programmi e/o schede hardware.

Ricordiamo inoltre che la routine, così com'è presentata, NON impedisce la copia fisica del dischetto originale nè il Reset, ed è quindi meglio dotare il proprio software di qualcosa di simile per impedirne totalmente la copia.

```
100 REM * * * * * * * * * * * * * * * * * *
110 REM *   ANTI-CARTRIDGE PROTECTION    *
120 REM * L. EMILITRI            * V 1.02 *
130 REM * * * * * * * * * * * * * * * * * *
140 :
150 REM * QUI INIZIA IL PROGRAMMA CHE *
160 REM * * DEVE ESSERE PROTETTO... * *
170 :
180 REM * * *    MOLTIPLICAZIONI    * * *
190 :
200 REM * *   DEFINIZIONE VARIABILI   * *
210 CP = 0            :REM NON E' COPIA
220 A=0 : B=0 : C=0 :REM VAR. CALCOLI
230 CM = 1            :REM NON C'E' ERRORE
240 A$ = ""           :REM TEMP. INPUT
250 :
260 GOSUB500 : REM >> ATTIVAZIONE! <<
270 :
280 REM * *   CORPO DEL PROGRAMMA    * *
290 PRINT CHR$(147) "MOLTIPLICAZIONI"
300 PRINT : INPUT "VALORE DI A...";A
310 INPUT "VALORE DI B...";B
320 C = (A*B)*CM
330 PRINT : PRINT "A * B =";C
340 :
350 REM *  CONTROLLA SE E' ORIGINALE   *
360 GOSUB 690
370 IF CP <> 0 THEN CM = RND(0)
380 :
390 PRINT : PRINT "PREMI UN TASTO..."
400 GET A$: IF A$="" THEN GO TO 400
410 GO TO 290
420 END : STOP : END : STOP : END : STOP
430 :
440 REM * * * * * * * * * * * * * * * * * *
450 REM * ROUTINES PER LA SCRITTURA E *
460 REM * IL SUCCESSIVO CONTROLLO  DI *
470 REM * UN TESTO NELLA RAM DEL 1541 *
480 REM * * * * * * * * * * * * * * * * * *
490 :
500 REM * *   DEFINIZIONE VARIABILI   * *
510 MS$="PASSWORD" :REM CHIAVE D'ACCESSO
520 MW = 193          :REM INIZIO RAM
530 P$= ""            :REM STRINGA COMANDI
540 B$= ""            :REM STR. CONFRONTO
550 A = 0             :REM INDICE DEL LOOP
560 :
570 REM *  SCRITTURA DELLA  PASSWORD  *
580 REM *  NELLA RAM DEL DRIVE 1541.  *
590 CLOSE 15: OPEN 15,8,15
600 FOR A = 1 TO LEN(MS$)
610 : P$ = CHR$(MW+A-1) + CHR$(0)
620 : P$ = P$ + CHR$(1) + MID$(MS$,A,1)
630 : PRINT#15, "M-W" P$;
640 : P$=""
650 NEXT A
660 CLOSE 15
670 RETURN
680 :
690 REM * LETTURA NELLA RAM DEL  1541 *
700 REM * PER  CONTROLLARE SE  ORIGI- *
710 REM *  NALE O PIRATEGGIATO [!!!]  *
720 CLOSE 15: OPEN 15,8,15
730 FOR A = 0 TO LEN(MS$)-1
740 : P$ = CHR$(A+MW) + CHR$(0)
750 : PRINT#15, "M-R" P$;
760 : GET#15, A$: B$ = B$+A$
770 NEXT A
780 IF MS$ <> B$ THEN CP=1
790 B$=""
800 CLOSE 15
810 RETURN
```

ANCHE UN FILE SI CONVERTE

Un breve programma, in AmigaBASIC, trasforma un qualsiasi file Ascii in uno di formato Notepad

di **Daniele Paccaloni**

Il programma Notepad mette a disposizione 10 pagine (numerate da 0 a 9) nelle quali scrivere testi di qualsivoglia natura.

Se, però, esaminiamo il file di testo salvato su disco possiamo notare diverse cose:

• 1: Ogni ritorno carrello (cioè un Return) viene codificato con il codice Ascii $0A, che corrisponde, in Basic, al CHR$ (10); questo codice viene chiamato "Line Feed" o, più brevemente, LF.

• 2: Per segnalare il cambio di pagina viene usato il codice Ascii $0C, in Basic CHR$(12), chiamato "Form Feed" o "FF".

Il Notepad, editor molto primitivo e semplice, si limita a convertire i caratteri di tabulazione orizzontale, detti "HT", Ascii $09 = CHR$(9), in spazi.

Per chi ancora non lo sapesse, il TAB è quel largo tasto grigio situato sopra il CTRL che, premuto, permette di posizionare il cursore nella prossima posizione di tabulazione individuata con la formula...

x = x + 8 - (x MOD 8)

...in cui X è la corrente posizione orizzontale del cursore.

Normalmente le posizioni di tabulazione sono posizionate ogni 8 caratteri, ma in alcuni programmi sono riprogrammabili.

Passiamo ora ad elencare gli inconvenienti che capitano quando si carica un file Ascii, "normale", direttamente con Notepad:

• 1: Se il testo non può essere ospitato per intero nella pagina (e non viene fornito alcun carattere "Form Feed") viene "mozzato" e i caratteri rimanenti vengono persi poichè non è più possibile passare alla pagina successiva.

• 2: Se il Notepad incontra alcuni caratteri di controllo, tra i quali i frequenti TAB, li visualizza sotto forma di un indesiderato carattere in reverse.

Supponiamo, ora, di voler trasferire un programma sorgente in assembler nel Notepad.

Tutti i TAB diventano una "I" in reverse e il programma perde l'incolonnamento.

IL RIMEDIO

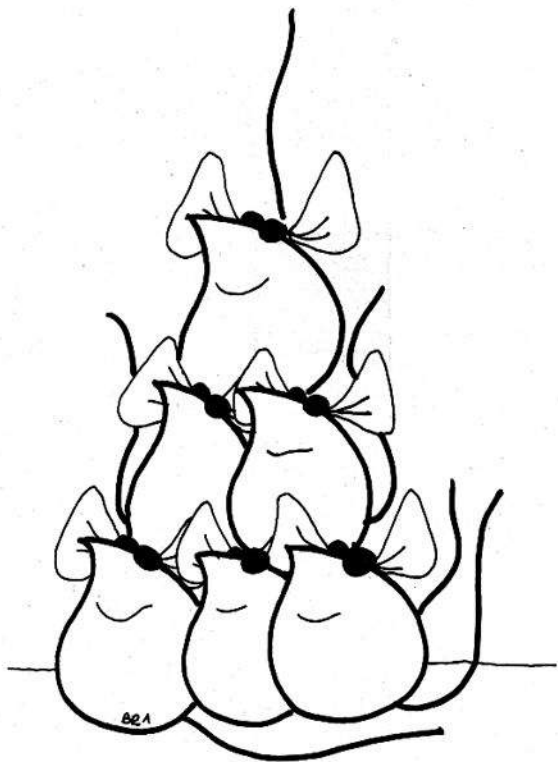
Il programma in Basic di queste pagine si limita a leggere un file Ascii normale, a sostituire i TAB con il corretto numero di spazi, e ad aggiungere, in coda al testo convertito, 10 caratteri di "Form Feed" per consentire di voltare pagina.

La dimensione dei TAB è riprogrammabile modificando il valore della costante "SPCinTAB" per potere convertire anche i testi nei quali il valore non sia 8.

La variabile T$() è dimensionata per contenere 1000 linee. Se il vostro testo ne contiene di più cambiate il 1000 in un numero che reputate sufficiente (occhio alla memoria).

Dato il RUN, il programma chiede il nome del file di testo da convertire (se si trova in una subdirectory o in un altro disco, specificatelo), quindi lo legge linea per linea visualizzandone la conversione, e infine chiede il nome con il quale dovrà essere salvato il file in formato Notepad.

L'unico inconveniente è che il primo carattere di ogni pagina non viene visualizzato, probabilmente perchè scambiato con il "Form Feed" finale della pagina precedente che noi... non abbiamo mai inserito; infatti i nostri "FF" sono tutti alla fine del testo. Prima di caricare il file convertito nel Notepad, allargate la sua finestra a tutto schermo.

Infine, per aumentare la velocità di conversione, compilate il programma se possedete, ovviamente, un idoneo compilatore.

```
' NOTEPAD CONVERTER 1.1 (converte un file ASCII in formato Notepad).
' 22/05/1989  D.Paccaloni

  DIM t$(1000)                 ' Dimensiona la matrice che conterra' il file
                               '   convertito (1000 linee MAX).

  LINE INPUT "Nome del file da convertire:";source$
  OPEN source$ FOR INPUT AS 1

  SPCinTAB = 8                 'Indica il numero di spazi con i quali
                               '   rimpazzare i TAB.

  FOR n=1 TO SPCinTAB          'Riempi s$ con il numero di spazi indicato
    s$=s$+" "                  '
  NEXT n                       '.
```

```
'     Routine per la lettura e conversione del file:

  lin=0                    'Inizializza il contatore di linee.

GetLine:
  col=0                    'Azzera il contatore di colonne.
  LINE INPUT #1,a$         'Legge una linea,
  t$(lin)=""               'annulla la stringa per la conversione.
  FOR x=1 TO LEN(a$)       'Ciclo per ricerca e rimpiazzamento dei TABs:
 char$=MID$(a$,x,1)                 'Prende un carattere,
 IF char$=CHR$(9) THEN              'Se il carattere e' un TAB allora:
   spaces=SPCinTAB-(col) MOD SPCinTAB '  Trova il giusto numero di spazi,
   t$(lin)=t$(lin)+LEFT$(s$,spaces)   '  lo rimpiazza con essi,
   col=col+spaces                   '  aggiusta contatore colonne
 END IF                             '.
 IF char$<>CHR$(9) THEN             'Se il carattere non e' un TAB:
   t$(lin)=t$(lin)+char$            '  Lo copia cosi' come e',
   col=col+1                        '  incrementa contatore colonne
    END IF                           ' .
  NEXT x                   'Prende il prossimo carattere...
  PRINT lin;": ";t$(lin)   'Visualizza la linea convertita.
  IF EOF(1) THEN Out       'Se il file e' terminato esce dal ciclo...
  lin=lin+1                'Altrimenti incrementa il contatore di linee,

  GOTO GetLine             'prende la nuova linea...

      La lettura e' terminata:

Out:
  CLOSE 1                  'Chiude il file.

      Salva il file convertito in formato Notepad

  CLS
  LINE INPUT "Nome del file convertito:";dest$
  OPEN dest$ FOR OUTPUT AS 1

  CLS
  PRINT "Linea:"
  FOR w=0 TO lin                       'Ciclo di scrittura:
    LOCATE 1,8                         '  Visualizza il numero di linea
    PRINT w;"    "                     '  .
    PRINT #1,t$(w)                     '  Scrive la linea su disco.
  NEXT w                               'Prende la prossima linea...

  FOR f=1 TO 10          'Ciclo per inserire i caratteri di fine pagina:
    PRINT #1,CHR$(12)    '  Inserisce un Form Feed (FF)
  NEXT f                 '.

 CLOSE 1                 'Chiude il file.

  PRINT "Conversione terminata."
```

Vuoi pubblicare un annuncio gratuito?

Commodore Computer Club ti offre la possibilità di pubblicare GRATUITAMENTE il tuo annuncio riguardo scambio o vendita di software, vendita o acquisto di apparecchi usati, ricerca di amici per fondare un club e così via. Invia l'annuncio in busta chiusa, affrancata secondo le norme vigenti, indirizzando a:

Commodore Computer Club - Viale Famagosta, 75 - 20142 Milano

Cerco "Niki Cartridge". Prezzo da stabilire. Telefonare a: Antonio Peruzzi 0774/678106. Il lunedì, mercoledì, venerdì dopo le ore 15.00.

La F.F. Company vende ottimi video-games e programmi per C=64-128 Vic 20. Tel. 0521/335336 - 0521/57407. Chiamare dalle 18.00 alle 19.00. 43100 Parma.

Vendo e scambio programmi per C=64 richiedere lista gratuita con oltre 4.000 titoli a: Gianni Mazzesi Via Cella 329 - 48020 S. Stefano (Ra). Tel. 0544/66507 o.p.

Vendo P.R.G. videoscrittura, grafica, musica, ingegneria, totocalcio, giochi, utility ed altro per C=64 - Tel. 095/515308 (Ct) - Mimmo.

Scambio: software ed esperienze su "Amiga-500". Posseggo programmi personali per studenti dell'ITG. Catalano Tommaso Via V.Calvanese, 34 80021, Afragola (NA) tel. 081/8696644

Vendo penna ottica con software su casetta a L. 30.000. Puoi usarla anche per i tuoi programmi per C64 & 128 sconti se comprate anche qualche gioco (richiedere lista) - Cicciotti Mario Via Vidimari, 41 67051 Avezzano (AQ) tel. 0863/552261

Amigàs! Programmi di produzione propria e mille altri! Info-disk L. 5.000 in busta chiusa: BAP (Josef), Via Appia Nuova, 244 - 00183 Roma

C64 compro: drive 1541 II originale L.120.000 max; mouse L.45000 scambio: software su disco/cas sette - Massimo Via 4 novembre, 70 Balestrate (PA) tel. 091/8786140

V.C.S. Vendo - scambio - giodhi per A 500 - Volta Sergio Via Voldomino, 32 Luino (VA) 21016 tel. 0332/530342 annuncio sempre valido

Vendo programmi per C64 e Amiga 500, L. 1.000 a facciata. Istruzioni in italiano, arrivi bisettimanali, anche abbonamenti. Oltre 12.000 prog. tutte le novità (Iss, Superman, Dragon Ninsa, e tanti altri). Per maggiori informazioni telefonare allo 06/9626429. Chiedere di Danilo.

Ricambio il favore a chiunque possa procurarmi una interfaccia Midi con software per C64. gradito anche solo schema e/o X RS 232-C. Tomasello Mario. Tel. 055/470676.

E' nato il Diegosoft (DGS)!!! Società che importa Hard/Soft per Amiga e lo rivende a prezzi stracciati!! Es.: programma (con disco compreso) L. 3.200 (in abbonamento L. 2.800)!!! Telefona o scrivi a: Diegosoft, V.le Cortemaggiore n. 12/2 - 93012 Gela (Cl). Tel. 0933/938404. Rispondi presto!!!

Cedo computer Commodore 128/64 - stampante MPS 801 trascinatori enciclopedia basic Curcio rilegata 8 volumi - 6 di basic e 2 Basic Advanced fare offerte a: C.P. 9 - 20036 Meda (Mi).

Vendo C64 + Drive 1541 orig. + registratore + adattatore telem. + copritastiera + 10 cassette + 20 dischi (ultime novità). A L. 600.000. Telefonare allo 0881/44098

Possiedi un Modem? Telefonami oggi, sono disposto a scambiare numeri di tel. di B.B.S. (banche dati). Tel. allo 011/2202898 Venaria (To). Telefonare a voce dalle 14.30/18.30 chiedere di Marchesello Roberto.

Vendiamo n.s. software originale (da noi prodotto) a bassissimi prezzi (solo zona Roma) rivolgersi al 3966764 - 3060401 ore pasti. Si garantisce massima serietà!!

Per C64 vendesi penna ottica grafica L. 15.000, Universal Cartridge EVM L. 30.000 (valore 80.000), Gelos 1 originale + manuale it. L. 15.000, tutti nuovi mai usati. Tel. dopo le 14.00 allo 06/9022289

Me.ar Soft Sistems. Vende/-scambia PRG X C64 su nastro scrivere a: Mear Soft Sistems c/o Mendolicchio Armando, Via G. Acquaviva n. 27 - 71100 Foggia

Ultime novità soft per Amiga con il solo costo del dischetto è possibile?! Certamente ma solo presso il S.A.C. (chiedere di Luca dalle 20-22). Per informazioni: 0362/501857

Vendo/scambio giochi e utilities per C64. Rispondo a tutti! Massima serietà! Tel. 0522/864874. Alessandro.

V.S.C. Vendo, scambio, compro per Amiga 500. Volta Sergio, Via Valdomino, 32 - 21016 Luino (Va). Tel. 0332/530342. Annuncio sempre valido.

Cerco manuali in italiano di: Pet speed - home word - the manager - disk demon - easy file - lode runner - diskmimic 4,5,6 - scrivere o telefonare a: Armando Natale - Via Salvo D'Acquisto, 10 93100 Caltanisetta - 0934/51206

Cerco utenti amiga 500 per scambio programmi - possiedo oltre 100 prg - scrivere a: Paolo di Maio P.zza Torino, 6 14100 Asti o telefonare 0141/215322

Cerco video monocromatico per C64 solo se buona occasione: Grottoli Gabriele Via C.Cattaneo, 59 20035 Lissone (MI) 039/462564

Vendo CBM + drive 1541C + 2 registratori + cartridge niki II + moltissimi programmi, sia su cassetta che su disco + portadischi tutto nuovo (L. 600.000) per informazioni rivolgersi allo 06/9786072 Colleferro (Roma) chiedete di Tora Simone

Cerco drive per commodore 64 in buone condizioni (quasi nuovo) modello 1571 compatibile a buon prezzo (contrattiamo) Lopez Gabriele Via Torino,39 Carmagnola tel. 011/9771985 (dalle 12 alle 14)

Compro - cambio - vendo - programmi per Amiga Attilio Manca Via G.Spagnolo, 15 98051 Barcellona (ME) tel 090/9703509

Superaffare!! Vendo CBM64 + datasette C2N + Light pen + duplicatore nas-nas + 2 joystik + tasto reset + numerosi giochi + voc. master a sole L. 300.000. Scrivere a: Coluissi Paolo Via Giuliano, 8 96014 Floridia (SR) Sicilia

Vendo corso di basic in cassette video basic per commodore vic 20 + raccoglitore cassette + 4 cassette giochi + cartuccia a L. 170.000 trattabili. Valenti Marco Via D.Alighieri, 33 Zanica (BG) tel. 035/672472

Vendo commodore C64 + monitor colore + 2 drive + stampante + registratore + espansione memoria + modem telefonico e molti programmi utility e giochi. Vero affare tel. 0923/982314

Chiedere informazioni su favoloso riduttore pronostici foto - enal8 - totip per plus/4 + 1541 + MPS802 con stampa schedine toto fino a 3000 colonne - Carmelo Mileto corso L.Razza, 14 - 89014 Oppido Mamertina (RC)

Vendo adattatore telematico 6499 per C64 C128 L. 70.000 corso video basic per C64 C128 su cassette o dischi L. 80.000 guida riferimento programmatore C64 L. 20.000 dischi 3/M nuovi e con giochi Livio Cavallo tel. 0171/84142

Desidero mettermi in contatto con esperti di C64 per esperienza cerco manuali in italiano scrivere o telefonare a: Salvatore Natale Via M. di Fatima,18 93100 Caltanisetta 0934/81079

Vendo per C64: test drive seul 88 Gijoe - Roadrunner - BMX - Infiltrator ed altri compro: combat school - super hang on - aliens bouble - dark castel - defende of crown telefonare allo 809034

Vendo commodore 64 + C2N registratore + circa 400 programmi (53 cassette) + corso basic (manuale + 2 cassette) Eugenio Romano Via 63, 35 C. Vetrano (TP)

Vendo scambio programmi e manuali per Amiga 500 scrivere a: Maxisoft Via tripoli,12 70123 Bari

Vendo scambio per C64 (su disco), a prezzi fantastici, per scrivi a: Toma Mario, Via G.Palmieri 73058 Tuglie (LE) tel. 0833/366689

Attenzione!!! Cerco i seguenti giochi: super - cycle - arkanoid per C64 (solo cassetta) telefonare ore seriali e chiedere di Judi tel. 0375/830580

Cerco utenti commodore64 disponibili scambiare Utility (data base, word processing, etc.) e perchè no, anche giochi educativi: Cuttone Diego Via E.Berlinguer,8 47038 Santarcangelo di R. Forlì

Vendo C128 + drive 1571 + stampante MPS 802 + vari programmi a lire 900.000 per informazioni 085-4681740 solo zona Pescara

Cerco drive 1571 in buono stato o un compatibile 1571 Telefonare 0331/791572 dalle 18.00 in poi e chiedere di antonio

Vendo C128 + drive L. 600.000 Marco Gandolfi - Via Davalli,12 43039 Salsomaggiore (PR) tel. 0524/70109 Importo trattabile

Vendo scambio giochi (solo su disco) C64 - Inviare e richiedere lista - Casella postale 109 22100 Como

Vendo corso di basic (video basic), per C64 in 20 cassette a L. 50.000 per informazioni telefonare al 02/4118391 e chiedere di Alex

Cerco utenti modem per collegamenti Manlio Milano tel. 011/7801766 telefonare la domenica

Vendo scambio programmi per C64 scrivere a: Mendolicchio Armando Via G.Acquaviva, 27 71100 Foggia tel. 0881/32683

Compro dischetti con giochi e/o programmi per C64 in buono stato e a prezzi modici - Scrivete o telefonate a: Elena Bolla Res. Fiori 332 cap. 20090 Milano (MI) tel. 02/26411012

Attenzione offro scambio tutti i migliori programmi per C64 Amiga IBM per totocalcio Lotto!!! Ascione Maurizio Via Panoramica, 10 80056 Ercolano (NA) tel. 081/7392240

Vendo giochi per C64 a L. 5000 il disco ho tutte le ultime novità ne possiedo anche su nastro richiedete la lista Amadori Marco Via Ferrari, 14 38068 Rovereto (TN) tel. 0464/413277

Software Amiga - Ultimissime novità sempre in arrivo prezzi ottimi telefonate: 079/238448 chiedere di Lorenzo (ore pasti)

Cerco in cosenza e provincia appassionati C64 per eventuale formazione di un club - Francesco tel. 21459

Vorrei molte informazioni sull'assembler e molti amici di Bit risposta assicurata - Trepiccione Marcello Via E.Fermi, 64 Casapulla Caserta cap. 81020

Vendo disco con prog. per toto totip pagamento c/assegno lire 12.000 Alberto Barboni C.P. 29 48100 Ravenna (anche su cassetta)

Cerco: testina per MPS 803 programmi per calcoli in C.A. e computo metrici per C64 Carmelo Giunta 0935/667846 Via A. Gramsci,92 94010 Assoro

Devi vendere un'Amiga o un monitor 1084? scrivi a: Umberto Lepore Via Colleoni, 14 24040 Bergamo tel. 035/907480

Compro espansione da 512 K per Amiga 500 (max L. 150.000) telefonare allo 049/8931294 e chiedere di Aronne

Cerco utenti commodore 64 vendo PRG a prezzi modici Cerco emulatore Cobol per 64 tel. 0974/822340 - Pasquale Liste gratuite max serietà

Cerco stampante grafica MPS 803 funzionante a prezzo modesto poss. in zona Milano Andreoni Damiano Via C.Colombo, 20 - Giaggiano (MI) 20083 telefonare o scrivere 02/9085800

Vendo giochi per C64 su dischi o cassette (pieni da ambo le parti) a L.3.000 cadauno - Tel. 06/7576706 Fabrizio Annuncio sempre valido

cerco utenti a 1000 con espansione 1 mega disposti vederla in ogni caso se la possedete telefonate a: Bragaggia Massimo 0422/543617

Vendo giochi per C64-128 solo su disco - Scrivere a:Cafaro Enzo Via Pirandello, 27 31021 Mogliano V. 041/5900921

Cerco mouse per C64 ad un buon prezzo tel. 055/6811126 Carlo (FI)

Scambio o vendo a prezzi stracciati programmi Amiga arrivi settimanali novità - Danilo Cornia -Via anziani,1 29100 Piacenza tel. 0523/65756

Vendo CBM64 quasi nuovo, completo di registratore + duplicatore di cassette + 150 giochi (out run, street fighter, ecc.), a L. 400.000 trattabili - tel. 02/2426622 chiedere di Giovanni

Cerco manuali Easy Script e SuperBase 64 (in italiano) anche fotocopiati, anche non originali cambio con soldi e/o prog. solo Milano. Marco tel. 02/4525020.

Vendo programmi per CBM 64 (giochi e util.) come: Hercules, Golf 3D, BuggyBoy, Tot 13, ecc. Invio gratis lista. Minimo 3 programmi L. 5.000. restuccia Marco - Via II strada C1 Nord Coop. Serenissima - cap. 81055 - S. Maria C.V. (CE) - tel. 0823-811417

Favoloso! Vendo 10 lezioni (su cassetta) Basic per il tuo CBM64 + per ogni acquirente in omaggio 2 programmi utility. Per informazioni rivolgersi allo 0883/45968 (ore pasti) - Chiedere di Davide (Trani - BA).

Vendo Commodore 128+Floppy disk 1541 + monitor a colori 1901, registratore e 2 Joystik. Michele Pavani Via Bentivoglio, 162 Ferrara 44100 tel. 0532/55533

Vendo per C=64 giochi e Utility di vario genere (su cass. e disco. Es: PrintMaster; Stampa Etichette etc.) a vari prezzi... convenienti!!! Tel. 02/6107734 ore pasti; chiedere di Paolo.

Vendo C64, drive 1541II, registratore, Joystick prorack di supporto, light pen enciclopedia Jackson, 500 programmi e riviste. Prezzo INTERESSANTISSIMO!! chiedere di Daniele allo 040/821117 0 scrivere a Scaggiante Davide Via Pinguente, 8 (TS)

CBM 128/64 scambio ultime novità offro e cerco tutti i migliori programmi con istruzioni tel. 0577/47054 - 53100 Siena - Cinci Giuliano - Pian dei Mntellini, 44

Per C64 vendo contrassegno dischi e cassette d'edicola con relativa rivista causa cambio sistema (vendo anche CCC n 59) - Scrivere a: Imperia - Oneglia Roberto Quaglia - Trav. Amoretti, 9 (IM)

Vendesi commodore 128 perfettamente funzionante (modo 64-128-CP/M) + drive 1571 doppia testina + registratore + 2 cartridges "final cartdge 2" - "fast load" + 50 disk con i migliori programmi + manuali - Alberici Diego Voghera (PV) tel. 0383/649138

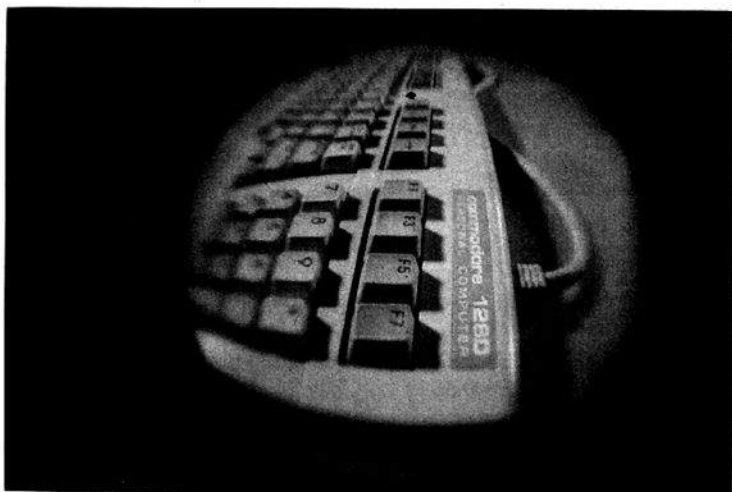
Acquisto espansione memoria esterna per Amiga 1000 - scambio/vendo programmi - Cottogni Gianni Via Strambino n 23 - 10010 Carrore (TO) - tel. 0125/712311 (18.00 - 21,00)

Vendo c128, Drive 1571, Reg. 1530, adatt. Tel. 6499, Cartucc. copiatore X 2 reg., The final Cartriage, raccolta 30 util. (Es. comp. Mod 740, equo canone, condominio ecc.) tutto per Lire 1.000.000. Vendo inoltre progr. per Amiga 500 a L. 4.000 compreso disco. Tel. 0962/902465 ore 19.30 in poi.

Attenzione!! La Top Game vende giochi come: Ghost'n Goblins, Out Run, Arkanoid, Enduro Racer, ecc. a L. 500 (!!) ciascuno. Richiedi subito la lista a: Marra Sandro, Via Gargiulo - 74029 Talsano (Ta). Max serietà.

Vendo Commodore 128D + Monitor 1901 + Stampante Riteman Cplus + Registratore Commodore C2N + Software 64/128/CPM a lire 1.500.000 trattabili. Tel. 035/570232

Iscrivetevi urgentemente al nostro club.
La quota di iscrizione è di sole L. 2.000 ed ai primi 15 soci la Deltron Soft offrirà loro una gradita sorpresa!!!
Gratis naturalmente!!!
Scrivete alla Deltron Soft - Via Adige, 28 - 04011 Aprilia (LT)

Il Master Soft Club offre ai soci migliaia di programmi di ogni genere su disco o su cassetta per C64/128 e Amiga, bollettino mensile, trucchi, novità, scambi, hardware vario ecc.
Iscrizione gratuita e massima serietà assicurata.
Nicola Gianno - Via Marsala, 351 - 91020 Rilievo (TP) - Tel. 0923/864559.

DIECI, CENTO, MILLE EDITOR

Una manciata di magiche locazioni per nuove ed insospettate prestazioni della tastiera del C/128

di **Domenico Pavone**

Come ben sanno i possessori dello sfortunato, ma pregevole, computer della fascia intermedia Commodore (quale sarà poi la fascia inferiore?), l'estrema facilità e comodità di utilizzo del C/128 è legata principalmente al suo editor di schermo, senz'altro uno dei più completi tra quelli in circolazione.

A scanso di equivoci, per editor di schermo si intende l'insieme hardware e software che consente, quando premiamo un tasto, di vederlo visualizzato sul monitor (o TV) che ci sta davanti, inclusi gli eventuali spostamenti del cursore, i caratteri semigrafici, e così via.

Ma comprende anche, e qui viene il bello, tutte le opzioni attivabili attraverso l'uso del tasto Escape, nonchè la gestione dei tasti programmabili (F1 / F8, Shift + Run / Stop, Help).

Tuttavia, nonostante la sovrabbondanza di performance già belle e pronte per l'utente, i progettisti del C/128 hanno (fortunatamente) pensato bene di non circoscrivere le potenzialità dell'editor, lasciandolo per così dire "aperto" ad eventuali interventi esterni.

Quanti di voi, per esempio, si sono chiesti cosa farsene del misterioso tasto Alt, abbandonato al suo destino lì in alto a sinistra della tastiera?

Oppure, ancora, come creare versioni personalizzate delle sequenze di escape? Tutto questo (ed altro), come vedremo soprattutto nella pratica, è molto più accessibile di quanto si pensi, grazie a dieci magiche locazioni di memoria, gestibili da tutte le configurazioni di banco: quelle da 820 ($334) a 829 ($33D).

Si trata di 5 cosiddetti "vettori" (vedi riquadro), ovvero coppie di locazioni contenenti ognuna un indirizzo, nel classico formato basso / alto.

Di queste speciali strutture a due byte, nel C/128 è presente un numero decisamente più elevato che nel suo predecessore a 64 Kappa e, in particolare, sono "vettorizzate" anche alcune routine dell'editor di schermo, allocato nella ROM dei banchi 14 e 15 da $C000 (49152) a $D000 (53248).

Proviamo a fargli visita, approfittando della comoda presenza del Monitor LM incorporato nella macchina.

Si prema dunque il tasto F8 e, una volta entrati in ambiente monitor, si digiti: D FC7B6 (e Return).

Apparirà, ovviamente, l'inizio del disassemblato di una routine dell'editor, la cui prima istruzione sarà un salto indiretto: JMP ($0334).

Per vedere dove effettivamente viene direzionato il flusso del programma, digitiamo ora M334, col solito Return finale.

I primi due valori esadecimali della riga apparsa, B9 e C7, rappresentano l'indirizzo in questione, ma, come già rimarcato, con il byte basso che precede il byte alto.

Il salto indiretto, quindi, corrisponde ad un JMP $C7B9, che fa proseguire la routine all'istruzione immediatamente successiva.

tabella 1

TUTTI I VETTORI DELL'EDITOR

nome	puntatori	contenuto
CTLVEC	\$334-\$335 (820-821)	\$C7B9 (51129)
SHFVEC	\$336-\$337 (822-823)	\$C805 (51205)
ESCVEC	\$338-\$339 (824-825)	\$C9C1 (51646)
KEYVEC	\$33A-\$33B (826-827)	\$C5E1 (50657)
KEYCHK	\$33C-\$33D (828-829)	\$C6AD (50861)

Salvo, ovviamente, nostro diverso parere.

UTILI DIROTTAMENTI

I vettori dell'editor di schermo, che potete vedere riassunti in tabella 1, si inseriscono tutti, in maniera diversa, nell'ambito di routine specializzate nella interpretazione di codici legati ai caratteri.

I puntatori appena esaminati (\$334 e \$335), in particolare, fanno capo alla sezione della routine di sistema Bsout, che si occupa di elaborare i valori Ascii compresi tra 0 e 31.

Come è possibile rilevare dal manuale del computer, questi codici non corrispondono alla visualizzazione sullo schermo di caratteri "normali", ma piuttosto a funzioni di controllo del colore, del cursore, e così via.

Quindi, per esempio, se premiamo Control e, contemporaneamente, il tasto "5", impartiamo un comando equivalente a Print Chr\$ (5), che imposterà il colore bianco per la scrittura sul video.

Essendo un codice Ascii inferiore a 32, questo viene "processato" (niente pendenze legali, per carità!) proprio dalla sezione della routine Bsout che inizia con il salto indiretto puntato da \$334 e \$335 (decimali 820 e 821).

POSTA C/128

MODI DI ESSERE

☐ Mi sono accorto, esplorando la memoria del mio C/128, che in modalità C/64 sono presenti, a partire dalla locazione 4106 (\$100A), gli stessi valori riscontrabili in "modo" 128, e cioè i caratteri Ascii associati ai tasti funzione (Graphic, Dload, eccetera). Nel passaggio da 128 a 64, non dovrebbe andare perduto ogni dato? E se così non fosse, sarebbe possibile trasferire semplici programmi basic o in linguaggio macchina tra i due computer?

(Andrea Lucarelli - Roma)

• Quando, con il C/128, ci si "sposta" da una modalità all'altra, si provoca l'attivazione della procedura di Reset relativa al computer cui si accede.

In altre parole, se ci si porta in modo 64 (con Go64 oppure premendo il reset con il tasto Commodore abbassato), prenderà il comando la routine di Reset del C/64; nel caso inverso sarà la procedura di Reset del 128 ad entrare in azione.

In entrambi i casi vengono inizializzate (e quindi modificate) solo le locazioni che servono al sistema in quel momento attivo per operare correttamente, come la pagina zero (le prime 255), vari puntatori, eccetera.

Tutto il resto, viene lasciato così com'è al momento del Reset.

In particolare, le locazioni da 4096 (\$1000) a 4351 (\$10FF) in Ram utente (non in Ram 1), riservate nel C/128 alla definizione dei tasti funzione, in modo 64 fanno parte della normale area adibita al basic, che inizia da 2049 (\$0801), e quindi non vengono alterate dal Reset.

Grazie a queste caratteristiche, ed a patto di conoscere bene la struttura dei due sistemi, un programma in linguaggio macchina può benissimo essere usato da entrambi i sistemi.

Addirittura, risulta molto comodo, dovendo assemblare brevi routines per il C/64, adoperare il Monitor del C/128 e poi passare al 64: la routine sarà sempre lì, ad aspettare.

Per quanto riguarda il basic, teoricamente valgono le stesse considerazioni, ma un programma sarebbe utilizzabile solo dopo complesse manipolazioni da linguaggio macchina.

Se vuoi provarci...

SCHERMATE GRAFICHE

☐ Come è possibile sovrapporre più schermate grafiche senza ricorrere ai troppo lenti Sshape e Gshape?

(Ferdinando Andretta - Caivano)

• Manipolare intere schermate grafiche con sufficiente velocità è compito praticamente impossibile senza ricorrere al linguaggio macchina.

Farlo con Sshape e Gshape, a parte la velocità, è proprio da suicidio.

Sul n. 51, a pagina 67, sono comunque illustrate alcune tecniche per simulare la contemporanea presenza di due schermate in alta risoluzione, facilmente implementabili anche programmando in basic.

ALTA VELOCITA'

☐ Si può far girare il C/64 incorporato nel C/128 a 2 Mhz, cosa che è possibile fare in modo 128?

(Massimiliano Nardi - Roma)

tabella 2

INDIRIZZI DA ANNOTARE

44803	($AF03)	Converte valore intero contenuto in Y (byte basso) ed accumulatore (byte alto), e lo deposita in FAC 1.
44806	($AF06)	Crea una stringa di caratteri che rappresentano il valore numerico contenuto in FAC 1. La stringa e' posizionata da $100 in poi, e termina con uno zero.
51126	($C7B6)	Elabora i codici Ascii inferiori a 32.
51202	($C802)	Processa codici Ascii superiori a 128.
51646	($C9BE)	Interpreta carattere dopo un Escape.
50525	($C55D)	Routine di scansione della tastiera.

* Tutti le locazioni si riferiscono al banco 15.

• Certamente.

Il controllo della frequenza del clock è affidato al bit 0 del registro 53296 ($D030) del VIC.

Se tale bit è azzerato, il processore girerà ad 1 MegaHertz, se invece è settato, viene implementato il Fast (2 Mhz).

Tale registro è raggiungibile anche in modo 64, per cui basterà una semplice...

Poke 53296, peek (53296) or 1

...per disporre della frequenza a 2 Mhz, mentre per tornare al modo normale occorrerà impartire...

Poke 53296, peek (53296) and 254

Operando in doppia velocità, però, lo schermo del C/64 "va in tilt", per cui non può essere adoperato.

Così come fa la routine di Fast del 128 (quando si lavora su 40 colonne), è opportuno quindi provvedere anche al cosiddetto Blank di schermo, ottenibile azzerando il bit 4 del registro 53265 ($D011).

Come esempio, si consideri questo breve listato, che si limita ad effettuare un ciclo a vuoto For...Next, controllandone la durata tramite la variabile TI, dopo aver attivato il blank di schermo e il modo fast.

```
10 poke 53265, peek (53265) and 239
20 poke 53296, peek (53296) or1
30 t=ti: for x=1 to 5000: next
40 t2=ti-t: print t2
50 poke 53296, peek (53296) and 254
60 poke 53265, peek (53265) or 16
```

Lanciandolo con un normale Run, e quindi attivando i 2 Mhz, la durata del loop risulterà di circa 160 jiffies (sessantesimi di secondo).

Se, invece, lo si manda in esecuzione con run 30 (velocità di 1 Mhz), il numero di jiffies salirà a circa 320.

In pratica, per usufruire della frequenza massima, sarà sufficiente inserire, al posto delle linee 30 e 40, tutte le procedure che si ritiene di dover velocizzare.

NUMERAZIONE DEL DRIVE

□ Perchè a pagina XXVI di Campus (numero 57) affermate "...mentre non lo consente all'utente del 1541...", riferendosi alla non possibilità di cambiare numero di device? Non è forse possibile via software o hardware, come riportato a pag. 45 del manuale del drive 1541?
(Antonio Raucci - S.Maria C.V.)

• Nell'articolo citato si fa riferimento ad una procedura particolare che consente, per esempio, con...

Open 15, 8, 15, "u0>" + chr$(9): close 15

...di modificare da 8 a 9 il numero di periferica del drive 1571.

Con il drive 1541 è sì possibile cambiare il numero di device come da manuale, ma, se si provasse ad usare integralmente la formula appena riportata, questa non verrebbe riconosciuta valida.

La frase citata, quindi, non esclude "in toto" la possibilità di modificare il numero di periferica, ma solo in rapporto all'uso di "U0>".

I motivi del fenomeno li puoi rintracciare leggendo più attentamente l'articolo in questione.

DOPPIO SALTO NON PROPRIO MORTALE

Com'è noto, il corretto funzionamento del computer è demandato alle routine in linguaggio macchina contenute nella memoria ROM del sistema.

Queste, dunque, non possono essere modificate in alcun modo.

Per consentire al programmatore di intervenire nello svolgimento di alcune di esse, sono stati inseriti, in alcuni punti, dei salti indiretti, ovvero delle istruzioni JMP($xxxx), che significano: "salta (GOTO) all'indirizzo contenuto nella locazione xxxx e nella successiva".

Normalmente l'indirizzo contenuto tra le parentesi corrisponde all'istruzione immediatamente successiva al salto.

Il motivo di questo tipo di procedura, è presto detto: le due locazioni contigue, che identificano un Vettore, si trovano sempre in RAM, e quindi sono liberamente manipolabili.

In pratica, basterà inserire in un vettore l'indirizzo (in formato low / hi) di una nostra routine, per fare in modo che questa venga eseguita al momento in cui viene attivata l'istruzione di salto.

Nella maggior parte dei casi la routine in questione deve concludersi con un ritorno all'indirizzo originario del vettore, ma non è una regola fissa. Diventa obbligatoria solo quando si ha a che fare con processi vitali per il sistema operativo, come nel caso tipico degli interrupt.

Più in dettaglio, al momento in cui viene chiamato in causa il vettore, il valore Ascii si trova immagazzinato nel registro accumulatore, pronto per essere trattato dal sistema... o da noi.

Con un opportuno "dirottamento" del vettore Ctlvec verso una nostra routine, infatti, possiamo modificare l'effetto dell'invio del codice, o addirittura creare nuove caratteristiche di editing.

Vediamo subito come applicare quest'ultima possibilità, della prima ce ne occuperemo tra breve, nell'ambito di un programma più elaborato.

Si copi, dunque, il brevissimo listato 1, che carica in memoria (da $1300 = dec. 4864 in poi) il programma LM che trova corrispondenza nel disassemblato 1.

Si tratta di una routine più che altro dimostrativa, in quanto si limita ad implementare un codice Ascii 1, che normalmente non sortisce alcun effetto, in modo da portare il cursore in fondo allo schermo.

Capito però il meccanismo di azione, lo stesso potrà poi essere applicato in maniera ben più proficua, per cui esaminiamo meglio il nostro nuovo CHR$(1).

Dopo il Run del listato 1, la nuova funzione può essere apprezzata digitando direttamente, o da programma, Print chr$(1), tenendo presente che, portando il cursore sull'ultima riga dello schermo, viene attivato lo scroll.

Per sfruttare al meglio questa risorsa, è dunque opportuno adoperare prima un Escape + M (= no scroll), o usare una sequenza tipo...

```
10 print chr$(1) "tutto okay"
20 print chr$(19): end
```

...che porta il cursore ad "home" dopo aver stampato la stringa sull'ultima riga.

PER QUALCHE CODICE IN PIU'

E veniamo al disassemblato 1, estremamente semplice nella sua brevità.

Quando il programma LM viene attivato con sys 4864, viene eseguito solo il blocco di istruzioni da 1300 ad 130A, che si limita ad installare nelle locazioni $334 e $335 (il nostro vettore, per intenderci) l'indirizzo $130B, ove inizia la routine vera e propria.

In pratica, ogni qualvolta viene inviato un codice allo schermo, il sistema è obbligato a passare prima attraverso il nostro "filtro", presente appunto da 130B in poi.

Qui viene controllato se il contenuto dell'accumulatore è uguale ad 1 (130B).

In caso negativo, si cede il controllo al sistema (130D), saltando all'indirizzo normalmente contenuto nel vettore (1318).

Se, invece, viene rilevata la presenza del codice Ascii 1, le istruzioni da 1310 a 1314 usano la routine Plot ($FFF0 del Kernal) per posizionare il cursore, inserendo il numero di riga in X e la colonna in Y.

Al fine di auspicabili smanettamenti, si tenga presente che i codici inferiori a 32 non utilizzati dal C/128 sono i seguenti: 1, 3, 4, 6, 16, 21, 22, 23, 25 e 26.

Quanto descritto finora può essere applicato anche ad un altro vettore dalle caratteristiche molto simili, quello identificato dalle locazioni $336 e $337.

Anch'esso viene richiamato da Bsout, ma, a differenza del precedente, punta alla sezione della routine di sistema che elabora i valori Ascii superiori a 128, comprendenti vari codici di controllo, nonchè tutti i caratteri semigrafici ottenibili tramite i tasti Shift e Commodore.

Inutile soffermarsi ulteriormente sull'argomento, se non per segnalare i codici liberi per eventuali nuove implementazioni: 128, 131 e 132.

Vedremo all'opera anche questo vettore, ma solo dopo aver esaminato il più "ghiotto" dei cinque, quello legato alle prestazioni del tasto Escape.

UN TRADUTTORE ISTANTANEO

La solita routine di sistema Bsout si occupa anche delle cosiddette Sequenze di Escape, in maniera non dissimile da quanto già visto.

Al tasto Escape, com'è noto, è associato il codice Ascii 27.

Quando l'editor riceve tale codice, passa il controllo ad una subroutine che inizia (a $C9BE) col solito salto indiretto, interessando stavolta il vettore $338-$339 (824-825), che punta all'indirizzo $C9C1 (51646).

Al momento del salto, nell'accumulatore è presente il codice Ascii associato al tasto premuto dopo Escape, che, se riconosciuto come valido dal sistema, attiva le routine del caso.

Anche questo vettore è manipolabile senza problemi, e possiamo subito vedere come, riferendoci al listato 2 e relativo disassemblato 2.

Questa volta si è realizzato qualcosa di più impegnativo del precedente esempio, ma non per questo più difficile.

Si copi dunque il listato 2 e, dopo averlo salvato, lo si mandi in esecuzione con Run.

Se tutto è in ordine dovrebbe apparire una breve segnalazione sullo schermo, dopodichè il computer torna ad essere... quello di sempre.

Si pressi ora il tasto Escape, e quindi il tasto "1".

Da questo momento, i caratteri alfabetici della tastiera verranno riprodotti normalmente, mentre, digitando un qualunque codice di controllo o carattere semigrafico, ne verrà stampato il relativo Ascii, ma in formato utilizzabile da basic.

Come esempio, si prema Home: il cursore non si porterà in cima allo schermo, ma apparirà la stringa-messaggio "Chr$ (19)".

Premendo invece in sequenza Escape e "0", si tornerà alle normali condizioni di default, manovra che si rende obbligatoria prima di eventuali Run, o Print di codici Chr$.

La routine può risultare utile tanto per avere una percezione immediata del valore Ascii corrispondente ai caratteri non alfabetici, quanto durante la stesura di programmi basic.

Se, per esempio, pensate di inviare un vostro bel programmino in redazione, una delle regole da rispettare è il non infarcirlo dei poco chiari simboli in reverse che caratterizzano tutti i codici di controllo.

Sostituendoli invece con i relativi Chr$(x)...

Il disassemblato della routine, ampiamente commentato, non necessita di particolari chiarimenti, per cui limitiamoci a riassumerne l'algoritmo generale:

• Si modifica il vettore di Escape per indirizzarlo al punto 2.

• Viene controllato il contenuto dell'accumulatore: se corrisponde al valore ascii del carattere "1", si modificano i due vettori che controllano i codici inferiori a 32 e superiori a 128, dirottandoli al punto 4.

Se, invece, è riscontrata la pressione del tasto 0, si ripristina il valore di default dei vettori e si torna al basic.

• Se il tasto premuto non è compreso tra i due appena visti, si salta al normale indirizzo della routine di Escape ($C9C1).

• Viene vagliata, prima di tutto, l'eventuale pressione di Escape, altrimenti non si riuscirebbe più ad uscire dal "modo traduzione". In caso positivo, si esce a Bsout, ma ad un indirizzo ($C7F5) più "avanzato", per evitare un ulteriore test (stavolta da parte del sistema) sul tasto Escape. Essendo il vettore di Bsout già dirottato, il controllo su Escape continuerà a passare per il punto 2.

• Viene stampato sullo schermo "Chr$(" sfruttando il Kernal ($FFD2), quindi si utilizzano due particolari routine di sistema (vedi riquadro) per visualizzare il valore Ascii del tasto premuto.

• Viene stampata una parentesi chiusa, e si ritorna al basic.

TASTIERA A TUTTO CAMPO

Sulla base di quanto visto risulterà ancora più facile occuparsi degli ultimi due vettori del gruppo in esame.

Si tratta delle locazioni $33A-$33B e $33C-$33D (per indirizzi contenuti vedi tabella 1), collegate non più alla routine di sistema Bsout, ma a quella che si occupa della scansione della tastiera, chiamata ScnKey.

In effetti, dei due vettori, quello davvero utile è il secondo.

Al momento in cui sono chiamati in causa, in entrambi i casi sarà presente in accumulatore il codice del tasto premuto ma, con $33A-$33B, si tratterà del codice di tastiera, meno maneggevole del solito Ascii.

Manipolando invece il secondo vettore, i vantaggi sono molteplici.

Anzitutto, si avrà a disposizione non solo il codice Ascii del tasto premuto (in accumulatore), ma anche la condizione dei tasti speciali (in registro X), prelevata dalla locazione 211.

In pratica, a seconda del tasto "speciale" premuto, in X sarà possibile rintracciare i seguenti valori...

Shift = 1
Commodore = 2
Control = 4
Alt = 8
Caps Lock = 16

... tra loro cumulabili.

La contemporanea pressione di Shift e Alt, per esempio, produrrà un valore 9, e così via.

Ecco che, in definitiva, diventa semplice una implementazione come quella proposta con la routine illustrata dal disassemblato 3, allocabile in memoria tramite il listato 3.

Attivata con Sys 4864 (già presente nel listato basic), si disporrà di una nuova risorsa collegata al tasto Alt: se premuto assieme ad "L", visualizzerà sullo schermo una linea di 40 caratteri.

Con la stessa tecnica già vista in precedenza, tale risultato è garantito dal dirottamento del vettore Keychk (1300 130A del disassemblato), mentre la breve routine di gestione si limita a testare il contenuto di X e, nel caso corrisponda alla pressione di Alt, controlla se sia stato adoperato il tasto "L" (130E-1314).

Qualora il riscontro abbia dato esito positivo, viene stampato (ancora tramite la routine del Kernal $FFD2) per 40 volte il carattere ascii 76, corrispondente al trattino ottenibile con Shift + asterisco.

La lunghezza della linea può essere modificata alterando il valore di X in 1316 direttamente da monitor, o da basic con una semplice Poke 4887, x.

Per concludere, è d'obbligo citare un'ul-

DARE I NUMERI

Un problema frequente per chi programma in Linguaggio Macchina, è rappresentato dalla visualizzazione di numeri sullo schermo.

Conoscendone a priori il valore sarà sufficiente utilizzare i codici Ascii corrispondenti alle singole cifre del numero in questione, ma non sempre le cose sono così semplici.

Come esempio, si consideri il disassemblato 2 di queste pagine: giunto all'istruzione 134F, deve visualizzare il codice Ascii del tasto premuto, presente in accumulatore dopo il PLA di 134F.

Supponendo, per esempio, che il contenuto di A sia uguale a 19, non è possibile inviarlo direttamente sullo schermo, se ne otterrebbe solo un "home" del cursore, chr$(19)="home".

Volendo, invece, stampare proprio un "19", o qualunque altro valore, è indispensabile ricorrere ai cosiddetti Accumulatori in Virgola Mobile, più semplicemente chiamati FAC (=Floating Accumulator).

Il sistema ne comprende due (FAC1 e FAC2), di solito utilizzati per effettuare calcoli matematici ma, per i nostri scopi, ne basta solo uno, FAC1.

In particolare, esiste una routine, con ingresso all'indirizzo $AF06 (44806), che converte il numero contenuto in FAC1 nella stringa di caratteri che lo rappresenta.

Quindi, se noi infiliamo il 19 in FAC1 e richiamiamo la routine in questione, ne otterremo i due caratteri Ascii corrispondenti ad "1" e "9", presenti nelle locazioni di memoria da $100 (256) in poi, con uno zero che indica la fine della stringa, e la prima locazione riservata al segno.

FAC1 (come del resto FAC2), però, può contenere valori solo nella particolare notazione in virgola mobile.

Per inserirvi un numero, è necessario dunque abilitare un'altra Routine: "targata" $AF03 (44803), che, molto semplicemente, si limita a depositare in FAC1, nella sua corretta notazione, il valore intero contenuto in Y (byte basso) ed Accumulatore (byte alto) al momento del suo richiamo.

L'insieme delle procedure descritte, è messo in pratica alle istruzioni da 1350 a 1360 del disassemblato 2.

tima possibilità legata al vettore $33C-$33D.

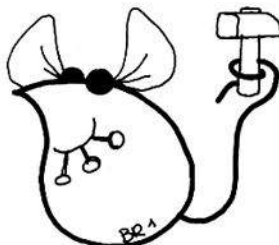
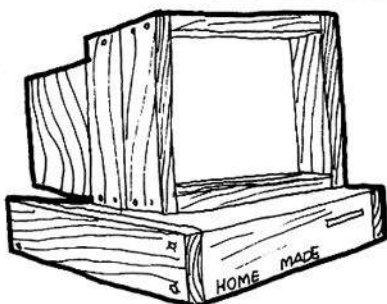
L'indirizzo in esso contenuto, è immediatamente a ridosso della sezione di Scnkey che controlla i tasti programmabili. Volendo escluderli, assegnando ad F1-F8 gli stessi codici Ascii che normalmente li rappresentano in modo C/64, basterà modificare il byte basso dell'indirizzo (in $33C) con una semplice Poke 828, 183. Con Poke 828,173 verranno invece ristabilite le condizioni di default.

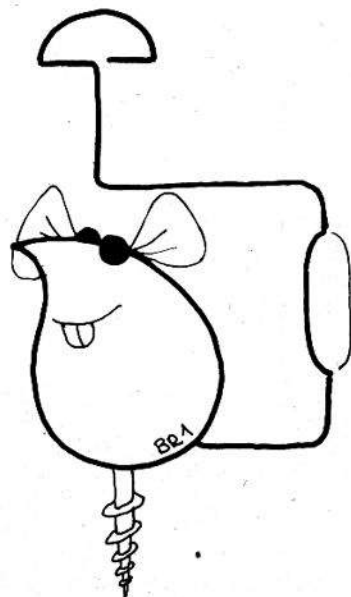
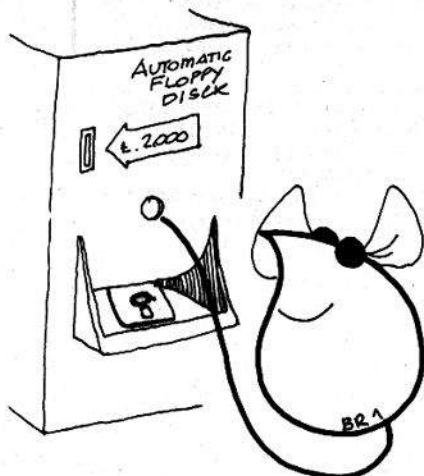
Un'altra Poke interessante, legata però al vettore $33A-$33B, può escludere completamente la tastiera: Poke 826, 92.

Se impartita in modo diretto, l'unico modo per continuare ad utilizzare il computer passa per un reset di sistema.

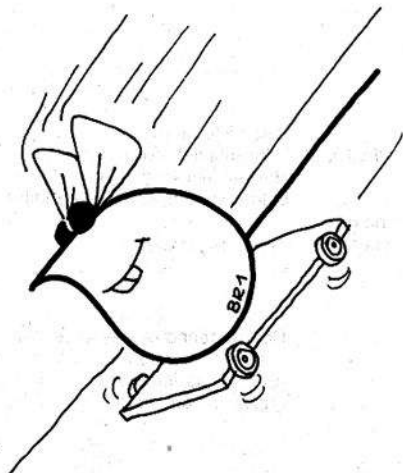
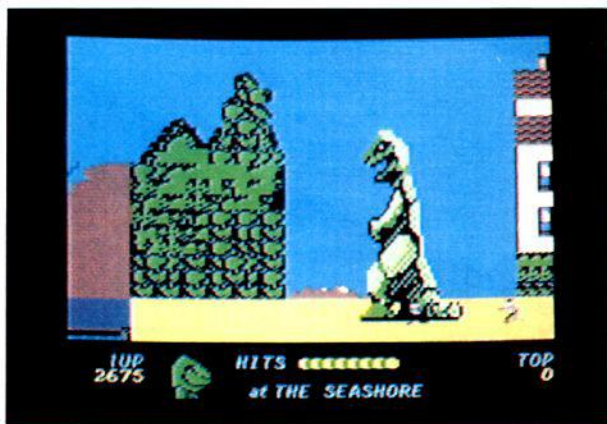
Inutile aggiungere che, ampliando il numero di implementazioni legate alle tecniche fin qui esposte, è possibile espandere quasi all'infinito le capacità della tastiera.

Fatene buon uso.

```
10 REM ----------------------------
15 REM L I S T A T O   1   (C/128)
20 REM ----------------------------
25 :
30 FORX=0TO26:READA:POKE4864+X,A
35 B=B+A:NEXT:IFB<>2713THEN45
40 SYS4864:PRINT"CHR$(1) ATTIVO":END
45 PRINT"ERRORE NELLE LINEE DATA!"
50 :
55 DATA 162,019,160,011,140,052,003
60 DATA 142,053,003,096,201,001,208
65 DATA 009,024,162,024,160,000,032
70 DATA 240,255,096,076,185,199
80 END
```

```
10 REM-------------------------------------------------
15 REM                    L I S T A T O   2     (C/128)
20 REM-------------------------------------------------
25 FAST:SCNCLR
30 FORX=0TO119:READA:POKE4864+X,A:B=B+A:NEXT
35 IFB<>13103THENPRINT"ERRORE NEI DATA!":END
40 PRINT"[ESCAPE] 1 - ATTIVA TRADUZIONE CODICI"
45 PRINT"[ESCAPE] 0 - DISATTIVA TRADUZIONE"
50 SLOW:SYS4864
55 DATA 169,011,162,019,141,056,003,142,057,003,096,201,049,208
60 DATA 002,240,026,201,048,208,087,162,199,160,185,140,052,003
65 DATA 142,053,003,162,200,160,005,140,054,003,142,055,003,208
70 DATA 064,162,019,160,061,140,052,003,140,054,003,142,053,003
75 DATA 142,055,003,208,046,201,027,240,046,072,162,000,189,114
80 DATA 019,240,006,032,210,255,232,208,245,104,168,169,000,032
85 DATA 003,175,032,006,175,162,000,189,001,001,240,006,032,210
90 DATA 255,232,208,245,169,041,032,210,255,096,076,193,201,076
95 DATA 245,199,067,072,082,036,040,000
98 END
```

```
10 REM ------------------------------
15 REM  L I S T A T O   3   (C/128)
20 REM ------------------------------
25 :
30 FORX=0TO32:READA:POKE4864+X,A
35 B=B+A:NEXT:IFB<>4542THEN45
40 SYS4864:PRINT"[ALT]+L ATTIVO":END
45 PRINT"ERRORE NELLE LINEE DATA!"
50 :
55 DATA 162,019,160,014,140,060,003
60 DATA 142,061,003,096,076,173,198
65 DATA 224,008,208,249,201,076,208
70 DATA 245,162,040,169,192,032,210
75 DATA 255,202,208,250,096
80 END
```

```
----------------------------------------
              DISASSEMBLATO  1
========================================

1300  LDX #$13   Dirotta vettori
1302  LDY #$0B   del salto indiretto
1304  STY $334   all' inizio della
1307  STX $335   routine (130B) e
130A  RTS        torna al basic.
----------------------------------------
130B  CMP #$1    Inviato codice 1?
130D  BNE $1318  Se no, esce a Bsout.
130F  CLC        Carry azzerato.
1310  LDX #$18   Riga 25.
1312  LDY #$00   Colonna 0.
1314  JSR $FFF0  Routine kernal Plot.
1317  RTS        Torna al basic.
----------------------------------------
1318  JMP $C7B9  Uscita a Bsout normale.
```

```
------------------------------------------
        DISASSEMBLATO  2
------------------------------------------

1300  LDA #$0B     Inserisce indirizzo
1302  LDX #$13     130B (in modo low/hi)
1304  STA $338     nel vettore di salto
1307  STX $339     a routine di Escape.
130A  RTS          Ritorna al basic.
------------------------------------------
130B  CMP #$31     Tasto "1" premuto?
130D  BNE $1311    Se no, salta a 1311.
130F  BEQ $132B    Se si', salta a 132B.
1311  CMP #$30     Taso "0" premuto?
1313  BNE $136C    Se no, esce a Escape.
------------------------------------------
1315  LDX #$C7     Vengono ripristinati
1317  LDY #$B9     i valori di default
1319  STY $334     (in formato basso/
131C  STX $335     alto) nei due
131F  LDX #$C8     vettori di salto
1321  LDY #$05     indiretto a BSOUT:
1323  STY $336     C7B9 in $334-$335 e
1326  STX $337     C805 in $336-$337.
1329  BNE $136B    Ritorna al basic.
------------------------------------------
132B  LDX #$13     Si inizializzano i
132D  LDY #$3D     due puntatori a
132F  STY $334     BSOUT in modo da
1332  STY $336     farli puntare ad
1335  STX $335     inizio di nostra
1338  STX $337     routine (133D).
133B  BNE $136B    Ritorna al basic.
------------------------------------------
133D  CMP #$1B     Premuto tasto Esc.?
133F  BEQ $136F    Se si', esce a Bsout.
1341  PHA          Salva tasto in stack.
1342  LDX #00      Legge caratteri
1344  LDA $1372,X  ascii di "chr$(".
1347  BEQ $134F    Se carattere=0, salta.
1349  JSR $FFD2    Li stampa su video.
134C  INX          Incrementa indice X
134D  BNE $1344    e continua il loop.
------------------------------------------
134F  PLA          Preleva tasto premuto
1350  TAY          e lo trasferisce in Y.
1351  LDA #$00     Byte alto ascii tasto.
1353  JSR $AF03    Valore ascii in FAC1.
1356  JSR $AF06    FAC1 diventa stringa.
1359  LDX #$00     Legge stringa da
135B  LDA $101,X   indirizzo $100 fino
135E  BEQ $1366    ad incontrare uno zero
1360  JSR $FFD2    e la stampa.
1363  INX          Ripete il ciclo per
1364  BNE $135B    tutti i caratteri.
1366  LDA #$29     Stampa su video una
1368  JSR $FFD2    parentesi chiusa.
136B  RTS          Ritorna al basic.
------------------------------------------
136C  JMP $C9C1    Uscita a Escape.
136F  JMP $C7F5    Uscita a Bsout.
------------------------------------------
1372  .BYT 'CHR$(' Stringa di 5 caratteri.
1377  .BYT 0       0 - fine stringa.
```

```
------------------------------------------
        DISASSEMBLATO  3
------------------------------------------

1300  LDX #$13   Dirotta vettori della
1302  LDY #$0E   scansione di tastiera
1304  STY $33C   all' inizio della
1307  STX $33D   routine (130E) e
130A  RTS        torna al basic.
------------------------------------------
130B  JMP $C6AD  Scansione normale.
------------------------------------------
130E  CPX #$08   Tasto Alt premuto?.
1310  BNE $130B  Se no,esce a Scansione.
1312  CMP #$4C   Tasto "L" premuto?
1314  BNE $130B  Se no,esce a Scansione.
1316  LDX #$28   Indice X = 40.
1318  LDA #$C0   Codice Ascii 76.
131A  JSR $FFD2  Lo stampa sullo schermo.
131D  DEX        X = X - 1.
131E  BNE $131A  Continua fino ad X=0.
1320  RTS        Ritorna al basic.
```

(a cura di **Andrea Ciaramella**)

THE MUNCHER
(Commodore 64)

Siete stanchi di fare gli eroi al servizio dell'umanità e volete fare qualcosa di sufficientemente malvagio? *The Muncher* è quello che fa per voi.

Infatti in The Muncher il giocatore impersona un dinosauro di enormi dimensioni che se ne va in giro per distruggere tutto ciò che trova sulla sua strada.

Tra le numerose vittime del Muncher ci sono militari, carri armati, elicotteri, povere donne che portano una carrozzina (tipo *Corazzata Potijomkin*) e così via. I livelli da superare sono 10 e comprendono: la base militare, il mare, la città Nintendo, Tokyo... il resto sta a voi scoprirlo.

Girovagando per le città il nostro caro dinosauro troverà anche delle uova (naturalmente della sua specie) e dei contenitori radioattivi dislocati in punti precisi del livello. Gli usi che possiamo fare di questi oggetti sono i seguenti: si possono mangiare i contenitori radioattivi (guadagnando energia) oppure mettere le uova nel contenitore, in modo da esser certi che, dopo la vostra morte, ripartirete proprio dall'ultimo punto nel quale avete depositato l'uovo.

La grafica di The Muncher è abbastanza curata e le dimensioni del rettile arrivano ad occupare i due terzi dello schermo (cosa abbastanza inusuale per il nostro vecchio C/64).

L'unico appunto che possiamo fare a The Muncher è quello di avere una scarsa longevità; non crediamo, infatti, che piacerà a molti l'idea di girovagare seminando distruzione e niente altro.

Comunque, se i vostri istinti bestiali sono stati repressi per lungo tempo è giunta l'ora di sfogarli.

ROCKSTAR ATE MY HAMSTER
(Commodore 64)

Quelli della Codemasters stavolta ritentano la fortuna con R.A.M.H. (sarebbe troppo lungo scrivere il nome per intero).

Nel gioco impersoniamo un manager in cerca di guadagno che, ereditando 50000 sterline da una vecchia zia, li investe per formare un gruppo rock.

Tutto inizia davanti allo schermo di un televisore nel quale appaiono i volti delle Rockstars in cerca di un manager con le rispettive caratteristiche; tra queste ne dobbiamo scegliere un numero variabile (da una a quattro) dopodichè inizierà la "vita" musicale del nostro complesso.

L'idea di gioco non è per niente malvagia; purtroppo non si può parlare meglio della realizzazione "concreta" dal momento che il gioco manca di spessore: dopo poco tempo si riduce ad alcune azioni meccaniche che danno sempre gli stessi risultati, all'infinito.

Dobbiamo dire, comunque, che il gioco è abbastanza divertente anche per l'alone di umorismo che lo circonda e riuscirà ad attrarvi per qualche tempo.

Per quanto riguarda la grafica, questa è sufficiente e lo stesso si può dire per il sonoro (ridotto a pochi Jingles che si alternano).

VIGILANTE
(Amiga)

Certo che in questi tempi le città si fanno sempre meno vivibili, la delinquenza dilaga e non si può mai uscire di casa con la convinzione di tornare interi o almeno vivi (abbiamo esagerato?): è proprio per questo motivo che, nelle grandi metropoli, è sempre più frequente la nascita di nuove organizzazioni di vigilantes.

I vigilantes sono dei normalissimi cittadini che si sostituiscono alla polizia cercando di autotutelarsi contro la delinquenza.

Nel gioco interpretate la parte di un vigilante in lotta contro una banda di rivali che hanno rapito una povera donna e la trasportano da una parte all'altra della città per mezzo di un furgone.

Il meccanismo di gioco è quello usuale dei "picchiaduro": il giocatore deve percorrere diversi livelli uccidendo tutti i nemici che incontra e, possibilmente, cercando di salvare se stesso.

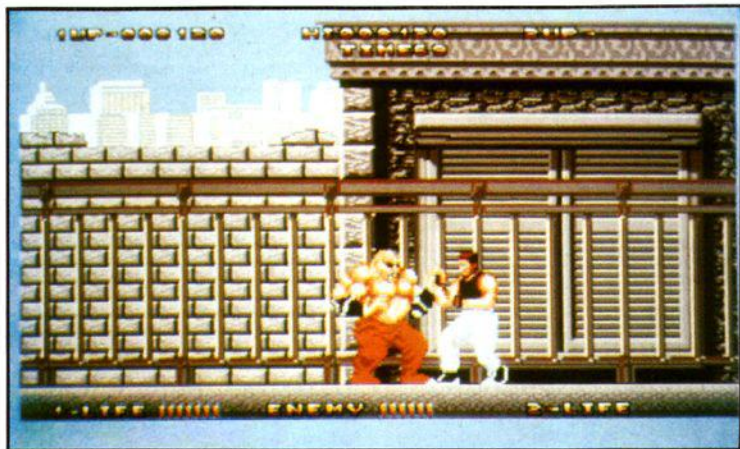
Il gioco è ambientato in diverse parti della città, tutte rappresentate in maniera abbastanza accurata e credibile. La grafica è nella media e l'azione di gioco piuttosto coinvolgente. Alla lunga Vigilante potrebbe diventare monotono, ma sappiamo benissimo che vi sono persone che vanno matte per questo tipo di giochi (altrimenti non si spiegherebbe il successo di giochi come Double Dragon o Dragon Ninja?) ed è proprio (e solo) a loro che consigliamo Vigilante.

DRAGON NINJA
(Amiga - C/64)

Vi avevamo già parlato in precedenza di Dragon Ninja per quanto riguardava la sua versione su Commodore 64 ma, dopo aver visto il corrispondente a 16-bit, abbiamo creduto opportuno esaminarlo una seconda volta.

Improvvisamente vi hanno avvisato che l'ex presidente degli stati uniti Ronald Reagan (Ronnie per gli amici) è stato rapito da una banda di malintenzionati. Decidete quindi, insieme ad un vostro amico, di andare allo sbaraglio per salvare il buon vecchio Ronnie dalle sgrinfie dei suoi nemici.

Questa storia sembra una versione riveduta e (poco) corretta di quella di Vigilante. Beh, ad essere sinceri lo è anche il gioco e non credo che ci metterete molto a capirlo. Parlando della sua struttura possiamo dire che anche Dragon Ninja è un picchiaduro a scorrimento orizzontale diviso in più livelli.

Cercando di compararlo con la versione per C/64 vengono alla luce due differenze sostanziali: la versione a 8-bit non dava la possibilità di giocare in due contemporaneamente (a differenza di quella a 16-bit) e l'azione di gioco della versione Amiga è decisamente migliore di quella per C/64.

LED STORM (Go!)
(C/64 - Amiga)

La *Go!* ha colpito: dopo continui alti e bassi qualitativi ha "sfornato" *LED STORM*, un gioco che di qualità ne ha da vendere.

Ci troviamo al comando di una futuristica macchina, completa di optionals, tra i quali *Max*, un computer di bordo, la possibilità di saltare (ricordate *Burning Rubber?*) e quella di trasformarsi in motocicletta in un batter d'occhio.

Ma della macchina cosa ne facciamo? Naturalmente partecipiamo ad una corsa. L'obiettivo della gara è quello di raggiungere la *Sky city* attraverso nove difficilissimi livelli.

Si parte dalla capitale, si arriva poi nel mezzo di un canon, su di un ponte che attraversa il mare di corallo, in una galleria, e... il resto sta a voi scoprirlo.

La difficoltà del gioco consiste nel raggiungere la fine di ogni livello senza consumare per intero il carburante a disposizione.

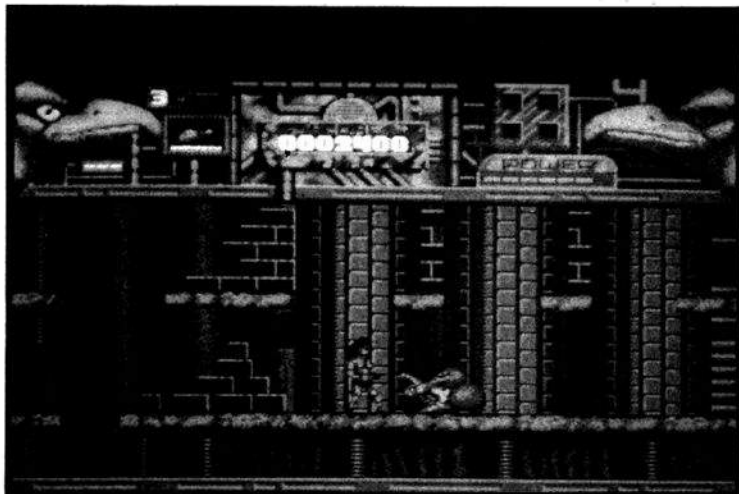
Per non restare a secco bisogna raccogliere taniche di benzina sparse lungo il percorso e collezionare i simboli *"E"* sganciati da astronavi che passano sulla vostra testa di tanto in tanto.

Ulteriori problemi sono rappresentati da macchine pronte a tutto pur di distruggervi, da interruzioni nei ponti e altre simili amenità.

Alla fine di ogni livello *Max* vi informerà sull'andamento della partita ed, eventualmente, aggiungerà punti bonus a quelli fino a quel momento accumulati.

Graficamente, pur non eccellendo, *Led Storm* compie bene il proprio dovere e ciò vale anche per il sonoro.

La cosa che più impressiona è la frenesia dell'azione, dovuta al gran numero di nemici ed allo scorrimento verticale, davvero veloce.

HAWKEYE
(Amiga - Commodore 64)

E' passato ormai molto tempo da quando *Hawkeye* è apparso sugli schermi del Commodore 64; ora la Thalamus lo ha convertito par Amiga.

In breve la storia è questa: siamo nell'era postnucleare e il mondo, prima abitato da uomini, è ora popolato di strane creature, per giunta realmente malvagie. Gli uomini, non potendo sopravvivere all'atmosfera radioattiva della superficie, sono stati costretti a rifugiarsi nel sottosuolo e, per combattere i loro nemici, hanno progettato un droide per metà umano e per metà robot capace di resistere a qualsiasi condizione ambientale.

Il droide è guidato dagli uomini per mezzo di un telecomando a distanza. Naturalmente i possessori del telecomando siete proprio voi ed è a voi che è affidato il compito di comandare il droide.

Nel vostro viaggio incontrerete moltissime creature che comprendono dinosauri, gorilla, grossi insetti, e così via e tutti quanti fermamente decisi a distruggervi (come, del resto, in tutti i giochi del genere).

La versione a 8-bit di Hawkeye ebbe un successo strepitoso dovuto all'incredibile qualità grafica e alla cura prestata ad ogni minimo particolare. Purtroppo non si puo dire lo stesso della versione per Amiga: infatti la grafica è pressocchè uguale a quella del 64 con una differenza: i colori utilizzati per la versione Amiga sono decisamente brutti (principale, grosso punto a sfavore). L'azione di gioco, inoltre, è identica a quella del 64 (con la differenza che quest'ultimo ha molte meno possibilità del suo fratello maggiore)

Hawkeye, insomma, è uno di quei giochi che fanno rimpiangere il buon vecchio C/64.

PHOBIA
(Commodore 64)

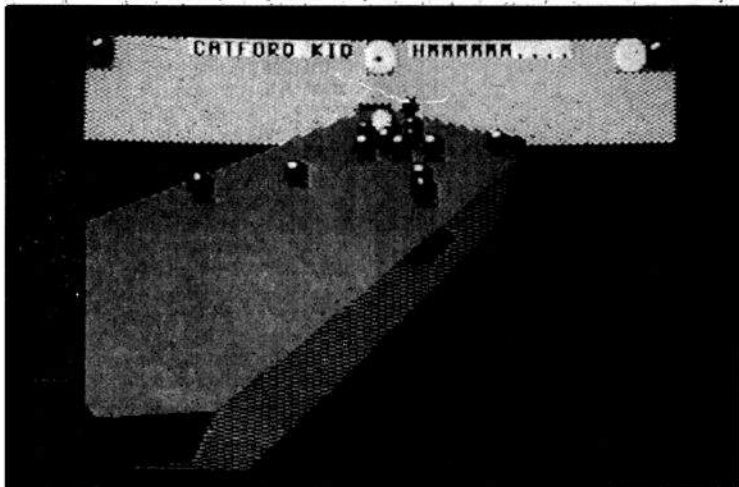
Certo che degli esperti di spara e fuggi come voi devono averne visti molti di nemici di ogni genere e tipo, dagli onnipresenti ufo ai megarobots, dai cammelli mutanti agli elicotteri, e così via. Ma in Phobia i nemici sono diversi da quelli di tutti gli altri giochi del genere dal momento che vi troverete a combattere contro le vostre stesse paure e vi possiamo assicurare che gli ideatori di Phobia non si sono certo risparmiati nell'inserire, nella loro creazione, mostri di ogni genere, come ragni giganteschi, scheletri, ghigliottine.

Il vostro compito, peraltro ovvio, consiste nel distruggere tutte le vostre paure prima di esserne sopraffatti. Il gioco è strutturato su molti livelli, a scorrimento orizzontale, ognuno rappresentante una differente paura e con un proprio mostro di fine livello.

Parlando della realizzazione tecnica di Phobia si può dire che mai un gioco potrà impressionare così favorevolmente, per quanto riguarda la sua realizzazione grafica. Infatti in Phobia tutto è curato fin nei minimi particolari, rendendo il gioco quanto di più giocabile e coinvolgente si sia mai visto.

Sfidiamo chiunque, persino l'Amiga, a riprodurre una grafica del tipo di quella di Phobia.

A proposito, al momento di andare in stampa abbiamo appena fatto in tempo a vedere la versione Amiga ma, ad essere sinceri, la prima impressione ricevuta è totalmente negativa.

3D-POOL
(Commodore 64)

Finalmente una simulazione di biliardo decente; anche se non si è particolarmente appassionati di biliardo, può far piacere svolgere una partita di tanto in tanto e fino ad ora non avevamo mai visto nessun titolo realmente valido.

La Firebird tenta di risollevare le sorti del biliardo computerizzato con 3D-Pool riescendoci benissimo. Le innovazioni che mettono 3D-Pool un gradino più in alto degli altri sono essenzialmente due: la sua realizzazione grafica in 3D grazie alla quale è possibile guardare il tavolo da qualsiasi angolazione, e l'intelligenza di livello quasi "umano" che caratterizza il giocatore comandato dal computer.

In campo vi sono tre tipi differenti di biglie: quelle rosse, quelle rosse a punti neri e la nera. I giocatori devono mandare in buca tutte le biglie del proprio colore e, per ultima (solo per ultima) quella nera. Colui che riuscirà ad "imbucare" la nera avrà vinto la partita.

Giocando contro il computer si partecipa ad un torneo ad eliminazione diretta, in cui ogni incontro si gioca al meglio delle tre partite (chi ne vince due passa il turno) e così via fino al termine del torneo.

E' possibile anche giocare contro un amico oppure posizionare le palline in campo per tentare tiri spettacolari. Nonostante la quantità di lavoro, svolto dal computer per l'aggiornamento grafico, l'azione di gioco è abbastanza veloce e divertente.

L'unica pecca di 3D-Pool è quella di impiegare troppo tempo per "pensare" alle mosse del giocatore computerizzato che, ad alcuni, potrebbe dar fastidio anche se a parer mio è un difetto irrilevante.

Ehi Tom, passami la stecca...

STORMLORD
(Commodore 64)

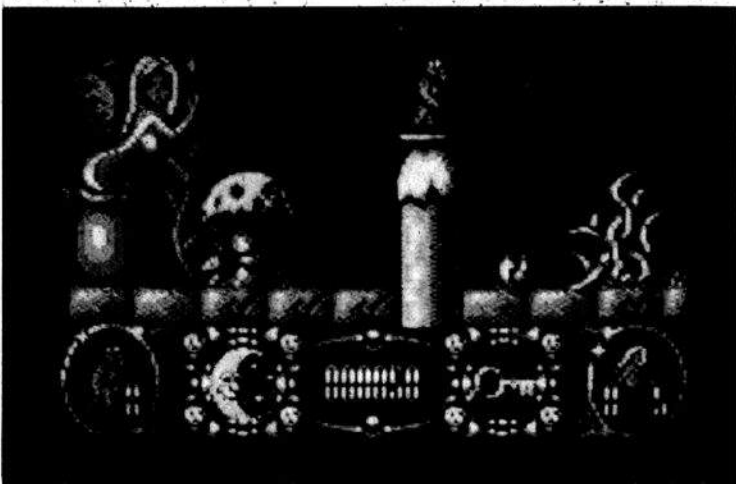
Cosa non si farebbe per le donne... Persino il nostro signore delle tempeste (Stormlord) che, a giudicare dal nome, dovrebbe essere un uomo spietato, quando si tratta di donne diventa la persona più gentile.

Da quando, infatti, le fate del mondo incantato in cui vive sono state imprigionate dalla strega malefica, il nostro Stormlord non trova più pace, è partito con la ferma convinzione di liberarle tutte (sperando di avere qualcosa in cambio?...).

Stormlord, l'ennesima "fatica" di Raffaele Cecco (personaggio notissimo tra gli "esperti" del settore) ha tutte le carte in regola per diventare uno dei giochi più venduti dell'89.

Scopo del gioco è quello (come ho già detto) di salvare le fate dalle grinfie della strega cattiva entro un limite di tempo prestabilito. Le fate sono cinque per ogni livello, i livelli sono quattro quindi, se la matematica non è un'opinione, complessivamente dobbiamo salvarne venti.

A questo punto ci chiederete quali sono i pericoli che si devono correre in Stormlord... mostri volanti alle pioggie acide, dallo scorrere del tempo alle porte sbarrate e... chi più ne ha più ne metta.

L'elogio più grande va sicuramente a coloro che hanno curato la parte grafica e sonora, asso nella manica di Stormlord. Tutti i fattori esposti in precedenza fanno di Stormlord uno dei giochi più divertenti ed accattivanti degli ultimi tempi.

QUANTO COSTA IL TUO COMMODORE

Amiga 2000 - L. 2.715.000

Microprocessore Motorola MC68000 - Clock 7.16MHz - Kickstart ROM - Memoria RAM: 1 MByte - 3 chip custom per DMA, Video, Audio, I/O - 5 Slot di Espansione Amiga Bus 100 pin Autoconfig™ - 1 Slot di Espansione 86 pin per Schede Coprocessore - 2 Slot di Espansione compatibili AT/XT - 2 Slot di Espansione compatibili XT - 2 Slot di Espansione Video - 1 Floppy Disk Drive da 3 1/2", 880 KBytes - Porta seriale RS232C - Sistema Operativo single-user, multitasking AmigaDOS - Compatibilità MS-DOS XT/AT disponibile con schede interne Janus (A2088 - A2286) - Monitor escluso

Amiga 500 - L. 995.000

Microprocessore Motorola MC68000 - Clock 7.16 MHz - Kickstart ROM - Memoria RAM: 512 KBytes - 3 Chip custom per DMA, Video, Audio, I/O - 1 Floppy Disk Driver da 3 1/2", 880 KBytes - Porta seriale RS232C - Porta parallela Centronics

Videomaster 2995 - L. 1.200.000

Desk Top Video - Sistema per elaborazini video semiprofessionale composto da genlock, digitalizzatore e alloggiamento per 3 drive A2010 - Ingressi videocomposito (2), RGB - Uscite Videocomposito, RF, RGB + sync -

Floppy Disk Driver A 1010 - L. 335.000

Floppy Disk Driver - Drive esterno da 3 1/2" - Capacità 880 KBytes - Collegabile a tutti i modelli della linea Amiga, alla scheda A2088 e al PC1

Floppy Disk Drive A 2010 - L. 280.000

Floppy Disk Drive - Drive interno aggiuntivo da 3 1/2" - Capacità 880 KBytes - Collegabile ad Amiga 2000

Hard Disk A 590 - L. 1.750.000

Hard Disk+Controller+RAM - Scheda Controller - Hard Disk da 3 1/2" 20 MBytes - 2 MBytes "fast" RAM - Collegabile all'Amiga 500

Scheda Janus A 2088 + A 2020 - L. 1.050.000

Scheda Janus XT+Floppy Disk Drive da 5 1/4", 360 KBytes - Scheda Bridgeboard per compatibilità MS-DOS (XT) in Amiga 2000 - Microprocessore Intel 8088 - Coprocessore matematico opzionale Intel 8087

A2286+A2020 - L. 1.985.000

Scheda Janus AT+Floppy Disk Drive da 5 1/4", 1.2 MBytes - Scheda Bridgeboard per compatibilità MS-DOS (AT) in Amiga 2000 - Microprocessore Intel 80287 - Clock 8 MHz - RAM: 1 MBytes on-board - Floppy Disk Controller on-board - Floppy Disk Driver disegnato per l'installazione all'interno dell'Amiga 2000 -

Scheda A2620 - L. 2.700.000

Scheda Processore Alternativo 32 bit - Scheda per 68020 e Unix - Microprocessore Motorola MC68020 - Coprocessore matematico Motorola MC68881 (opzionale MC68882)

Scheda A Unix - L. 3.250.000

Sistema Operativo AT&T Unix System V Release 3 - Per Amiga 2000 con scheda A2620 e Hard Disk 100 MBytes

Hard Disk A2092+PC5060 - L. 1.020.000

Hard Disk e controller - Hard Disk 3 1/2" ST506 - Capacità formattata 20 MBytes

Hard Disk A2090+2092 - L. 1.240.000

Hard Disk e controller - Hard Disk 3 1/2" ST506 - Capacità formattata 20 MBytes

Hard Disk A2090+A2094 - L. 1.900.000

Stesse caratteristiche del kit A2092 ma con disco da 40 MBytes

Espansione di memoria A2058 - L. 1.149.000

Espansione di memoria - Scheda di espansione per Amiga 2000 - Fornita con 2 MBytes "fast" RAM, espandibile a 4 o 8 MBytes

Scheda Video A2060 - L. 165.000

Modulatore video - Scheda modulatore video interna per Amiga 2000 - Uscite colore e monocromatica - Si inserisce nello slot video dell'Amiga 2000

Genlock Card A2301 - L. 420.000

Genlock - Scheda Genlock semiprofessionale per Amiga 2000 - Permette di miscelare immagini provenienti da una sorgente esterna con immagini provenienti dal computer

Professional Video Adapter Card A2351 - L. 1.500.000

Professional Video Adapter - Scheda Video Professionale per Amiga 2000 (B) - Genlock qualità Broadcast - Frame Grabber - Digitalizzatore - Include software di controllo per la gestione interattiva
(Disponibile da maggio '89)

A501 - L. 300.000

Espansione di memoria - Cartuccia di espansione di memoria da 512 KBytes per A500

A520 - L. 45.000

Modulatore RF - Modulatore esterno A500 - Permette di connettere qualsiasi televisore B/N o colori ad Amiga 500

A Scart - L. 28.000
Cavo di collegamento A500/A2000 con connettore per televisione SCART

Monitor a colori 1084 - L. 615.000
Monitor a colori ad alta risoluzione - Tubo 14" Black Matrix antiriflesso - Pitch 0.39 mm - Compatibile con Amiga 500/2000, PC (tutta la gamma), C64 e C128

Monitor a colori 2080 - L. 770.000
Monitor a colori ad alta risoluzione e lunga persistenza - Tubo 14" Black Matrix antiriflesso - Pitch 0.39 mm - Frequenza di raster 50 Hz - Compatibile con Amiga 500/2000, PC (tutta la gamma), C64 e C128

Monitor Monocromatico A2024 - L. 1.235.000
Monitor monocromatico a fosfori "bianco-carta" - Turbo 14" antiriflesso - (Disponibile da marzo '89)

PC60/40 - L. 8.930.000
Microprocessore Intel 80386 - Coprocessore matematico opzionale Intel 80387 - Clock 8 o 16 MHz selezionabile via software e da tastiera - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 funzioni - Sistema Operativo MS-DOS 3 2.1 - Interprete GW-Basic

PC60/40C - L. 9.180.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC 60/80 - L. 10.450.000
Microprocessore Intel 80386 - Coprocessore opzionale Intel 80387 - Clock 8 o 16 MHz selezionabile via software e da tastiera - Memoria RAM: 2.5 MBytes - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Floppy Disk Drive opzionale da 3 1/2", 1.44 MBytes - 1 Hard Disk da 80 MBytes - 2 Porte parallele Centronics - Mouse video EGA (compatibile MDA - Hercules - CGA). Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Ambiente Operativo Microsoft Windows/386 - Interprete GW-Basic

PC60/80C - L. 10.700.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC40/20 - L. 4.100.000
Microprocessore Intel 80286 - Coprocessore matematico opzionale Intel 80287 - Clock 6 o 10 MHz selezionabile via software, hardware o da tastiera - Memoria RAM: 1 MByte - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232 - Porta parallela Centronics - Scheda video AGA multistandard (MDA - Hercules - CGA) Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC40/20C - L. 4.350.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC 40/40 - L. 5.285.000
Microprocessore Intel 80286 - Coprocessore matematico opzionale Intel 80287 - Clock 6 o 10 MHz selezionabile via software, hardware o da tastiera - Memoria RAM: 1 MByte - 1 Floppy Disk Drive da 5 1/4", 1.2 MBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232 - Porta parallela Centronics - Scheda video AGA multistandard (MDA - Hercules - CGA) Emulazioni disponibili via hardware e software - Monitor monocromatico 14" - Tastiera avanzata 102 tasti con 12 tasti funzione - Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC40/40C - L. 5.535.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

1352 - L. 78.000
Mouse - Collegabile con Microsoft Bus Mouse - Collegabile direttamente a PC1, PC10/20 - III, PC40 - III

PC910 - L. 355.000
Floppy Disk Drive - Drive interno aggiuntivo da 3 1/2" per PC10/20-I-II-III - Capacità 360 o 720 KBytes selezionabile tramite "config. sys" - Corredo di telaio di supporto per l'installazione in un alloggiamento per un drive da 5 1/4" - Interfaccia identica ai modelli da 5 1/4"

PC1 - L. 995.000
Microprocessore Intel 8088 - 1 Floppy Disk Drive da 5 1/4" - Porta seriale RS232C - Porta parallela Centronics - - Monitor monocromatico 12" - Tastiera 84 tasti - Sistema Operativo MS-DOS 3.2 - Interprete GW-Basic

PCEXP1 - L. 640.000
PC Expansion Box - Box esterno di espansione per PC 1 - Alimentatore aggiuntivo incluso - Contiene 3 Slot di Espansione compatibili Ibm XT - Alloggiamento per Hard Disk da 5 1/4" - Si posiziona sotto il corpo del PC1 e viene collegato tramite degli appositi connettori

PC10-III - L. 1.965.000
Microprocessore Intel 8088 Clock 4.77 MHz 9.54 MHz (double) selezionabile via software e da tastiera - Memoria RAM: 640 KBytes - 2 Floppy Disk Drive da 5 1/4", 360 KBytes - Porta seriale RS232C - Porta parallela Centronics - Porta Mouse per Mouse Commodore 1352 (compatibile Microsoft Bus Mouse - Tastiera avanzata 102 con 12 tasti funzione Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC10-IIIC - L. 2.300.000
Stessa configurazione ma con monitor 14" a colori mod. 1084

PC20-III - L. 2.715.000
Microprocessore Intel 8088 - Clock 4.77 MHz 9.54 MHz (double) selezionabile via software e da tastiera - 1/4", 360 KBytes - 1 Hard Disk da 20 MBytes - Porta seriale RS232C - Porta parallela Centronics - Porta Mouse per Mouse Commodore 1352 (compatibile Microsoft Bus Mouse) - Tastiera avanzata 102 con 12 tasti funzione Sistema Operativo MS-DOS 3.21 - Interprete GW-Basic

PC20-IIIC - L. 3.050.000

Stessa configurazione ma con monitor 14" a colori mod. 1084

Nuovo C64 - L. 325.000

Nuovo Personal Computer CPU 64 KBytes RAM - Vastissima biblioteca software disponibile - Porta seriale Commodore - Porta registratore per cassette - Porta parallela programmabile -

C128D - L. 895.000

Personal Computer CPU 128 KBytes RAM espandibile a 512 KBytes - ROM 48 KBytes - Basic 7.0 - Tastiera separata - Funzionante in modo 128,64 o CP/M 3.0 - Include floppy disk drive da 340 KBytes

Floppy Disk Drive 1541 II - L. 365.000

Floppy Disk Drive - Floppy Disk Drive da 5 1/4" singola faccia - Capacità 170 KBytes - Alimentazione separata - Compatibile con C64, C128, C128D

Floppy Disk Dirve 1581 - L. 420.000

Floppy Disk Drive da 3 1/2" doppia faccia - Capacità 800 KBytes - Alimentazione separata - Compatibile con C64, C128, C128D

1530 - L. 55.000

Registratore a cassette per C64, C128, C128D

Accessori per C64 - 128D

1700 - Espansione di memoria - Cartuccia di espansione di memoria a 128 KBytes per C128 **- L. 170.000**

1750 - Espansione di memoria - Cartuccia di espansione di memoria 512 KBytes per C128 - **L. 245.000**

1764 - Espansione di memoria - Cartuccia di espansione di memoria a 256 KBytes per C64
Fornita di alimentatore surdimensionato - **L. 198.000**

16499 - Adattatore Telematico Omologato - Collegabile al C64
Permette il collegamento a Videotel, P.G.E. e banche dati - **L. 149.000**

1399 - Joystick - Joystick a microswitch con autofire - **L. 29.000**

1351 - Mouse - Mouse per C64, C128, C128D - **L. 72.000**

Monitor Monocromatico 1402 - L. 280.000

Monitor monocromatico a fosfori "bianco-carta" - Turbo 12" antiriflesso - Ingresso TTL - Compatibile con tutta la gamma PC

Monitor Monocromatico 1404 - L. 365.000

Monitor monocromatico a fosfori ambra - Turbo 14" antiriflesso a schermo piatto - Ingresso TTL - Compatibile con tutta la gamma PC - Base orientabile

Monitor Monocromatico 1450 - L. 470.000

Monitor monocromatico BI-SYNC a fosfori "bianco-carta" - Turbo 14" antiriflesso - Ingresso analogico e digitale - Doppia frequenza di sincronismo orizzontale per compatibilità con adattatori video MDA, Hercules, CGA, EGA e VGA

Monitor a colori 1802 - L. 445.000

Monitor a colori - Turbo 14" - Collegabile a C64, C128, C128D

Monitor monocromatico 1900 - L. 199.000

Monitor monocromatico a fosfori verdi - Turbo 12" antiriflesso - Ingresso videocomposito - Compatibile con tutta la gamma Commodore

Monitor a colori 1950 - L. 1.280.000

Monitor a colori BI-SYNC alta risoluzione - Turbo 14" antiriflesso - Ingresso analogico e digitale - Doppia frequenza di sincronismo orizzontale per compatibilità con adattatori video MDA, Hercules, CGA, EGA e VGA

Stampante MPS 1230 - L. 465.000

Stampante a matrice di punti - Testina a 9 aghi - 120 cps - Bidirezionale - 80 colonne - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia seriale Commodore e parallela Centronics - Compatibile con tutti i prodotti Commodore

MPS 1230R - L. 19.000

Nastro per stampante

Stampante MPS 1500C - L. 550.000

Stampante a colori a matrice di punti - Testina a 9 aghi - 130 cps - Bidirezionale - 80 colonne - Supporta nastro a colori o nero - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia parallela Centronics - Compatibile con la gamma Amiga e PC

MPS1500R - L. 37.000

Nastro a colori per stampante

Stampante MPS 1550C - L. 575.000

Stampante a colori a matrice di punti - Testina a 9 aghi - 130 cps - Bidirezionale - 80 colonne - Supporta nastro a colori o nero - Near Letter Quality - Stampa grafica - Fogli singoli e modulo continuo - Trascinamento a trattore e/o frizione - Interfaccia seriale Commodore e parallela Centronics - Compatibile con tutti i prodotti Commodore

I COMMODORE POINT

LOMBARDIA

Milano
• AL RISPARMIO - V.LE MONZA 204
• BCS - VIA MONTEGANI 11
• BRAHA A. - VIA PIER CAPPONI 5
• E.D.S. - C.SO PORTA TICINESE 4
• FAREF - VIA A. VOLTA 21
• FLOPPERIA - V.LE MONTENERO 31
• GBC - VIA CANTONI 7 - VIA PETRELLA 6
• GIGLIONI - V.LE LUIGI STURZO 45
• L'UFFICIO 2000 - VIA RIPAMONTI 213
• LOGITEK - VIA GOLGI 60
• LU - MEN - VIA SANTA MONICA 3
• MARCUCCI - VIA F.LLI BRONZETTI 37
• MELCHIONI - VIA P. COLLETTA 37
• MESSAGGERIE MUSICALI - GALLERIA DEL CORSO 2
• NEWEL - VIA MAC MAHON 75
• PANCOMMERZ ITALIA - VIA PADOVA 1
• SUPERGAMES - VIA VITRUVIO 38
• 68000 E DINTORNI - VIA WASHINGTON 91

Provincia di Milano
• GINO FERRARI CENTRO HI-FI - VIA MADRE CABRINI 44 - S. ANG. LODIGIANO
• F.LLI GALIMBERTI - VIA NAZIONALE DEI GIOVI 28/36 - BARLASSINA
• TECNOLUX - VIA PIETRO NENNI 5 - BERNATE TICINO
• OGGIONI & C. - VIA DANTE CESANA 27 - CARATE BRIANZA
• AL RISPARMIO - VIA U. GIORDANO 57 - CINISELLO BALSAMO
• GBC - V.LE MATTEOTTI 66 - CINISELLO BALSAMO
• CASA DELLA MUSICA - VIA INDIPENDENZA 21 - COLOGNO MONZESE
• PENATI - VIA VERDI 28/30 - CORBETTA
• EPM SYSTEM - V.LE ITALIA 12 - CORSICO
• P.G. OSTELLARI - VIA MILANO 300 - DESIO
• CENTRO COMPUTER PANDOLFI - VIA CORRIDONI 18 - LEGNANO
• COMPUTEAM - VIA VECELLIO 41 - LISSONE
• M.B.M. - C.SO ROMA 112 - LODI
• L'AMICO DEL COMPUTER - VIA CASTELLINI 27 - MELEGNANO
• BIT 84 - VIA ITALIA 4 - MONZA
• IL CURSORE - VIA CAMPO DEI FIORI 35 - NOVATE MIL.
• I.C.O. - VIA DEI TIGLI 14 - OPERA
• R & C ELGRA - VIA SAN MARTINO 13 - PALAZZOLO MIL.
• ESSEGIEMME SISTEMI SAS - VIA DE AMICIS 24 - RHO
• TECNO - CENTRO - VIA BARACCA 2 - SEREGNO
• NIWA HARD&SOFT - VIA B. BUOZZI 94 - SESTO SAN GIOV.
• COMPUTER SHOP - VIA CONFALONIERI 35 - VILLASANTA
• ACTE - VIA B. CREMIGNANI 13 - VIMERCATE
• IL COMPUTER SERVICE SHOP - VIA PADANA SUPERIORE 197 - VIMODRONE

Bergamo
•D.R.B. - VIA BORGO PALAZZO 65
• TINTORI ENRICO &C. - VIA BROSETA 1
• VIDEO IMMAGINE - VIA CARDUCCI c/o CITTA' DI MERCATO

Provincia di Bergamo
BERTULEZZI GIOVANNI - VIA FANTONI 48 - ALZANO LOMBARDO
• COMPUTER SHOP - VIA VITTORIO VENETO 9 - CAPRIATE SAN GERVASIO
• B M R - VIA BUTTARO 4/T - DALMINE
• MEGABYTE 2 - VIA ROMA 61/A - GRUMELLO
• OTTICO OPTOMETRISTA ROVETTA - P.ZZA GARIBALDI 6 - LOVERE
• COMPUTER POINT - VIA LANTIERI 52 - SARNICO
• A.B. INFORMATICA - STRADA STATALE CREMASCA 66 - URGNANO

Brescia
• MASTER INFORMATICA - VIA F.LLI UGONI 10/B

PROVINCIA DI BRESCIA
• MISTER BIT - VIA MAZZINI 70 - BRENO
• CAVALLI PIETRO - VIA 10 GIORNATE 14 BIS - CASTREZZATO
• VIETTI GIUSEPPE - VIA MILANO 1/B - CHIARI
• MEGABYTE - P.ZZA MALUEZZI 14 - DESENZANO DEL GARDA
• BARESI RINO &C. - VIA XX SETTEMBRE 7 - GHEDI
• INFO CAM - VIA PROVINCIALE 3 - GRATACASOLO
• "PAC-LAND" di GARDONI - CENTRO COM.le - LA CASA DI MARGHERITA D'ESTE - VIA GIORGIONI 21

Como
• IL COMPUTER - VIA INDIPENDENZA 90
• 2M ELETTRONICA - VIA SACCO 3

Provincia di Como
• ELTRON - VIA IV NOVEMBRE 1 - BARZANO'
• DATA FOUND - VIA A. VOLTA 4 -ERBA
• CIMA ELETTRONICA - VIA L. DA VINCI 7 - LECCO
• FUMAGALLI - VIA CAIROLI 48 -LECCO
• RIGHI ELETTRONICA - VIA G. LEOPARDI 26 - OLGIATE COMASCO

Cremona
• MONDO COMPUTER - VIA GIUSEPPINA 11/B
• PRISMA - VIA BUOSO DA DOVARA 8
• TELCO - P.ZZA MARCONI 2/A

Provincia di Cremona
• ELCOM - VIA IV NOVEMBRE 56/58 - CREMA
• EUROELETTRONICA - VIA XX SETTEMBRE 92/A - CREMA

Mantova
• COMPUTER CANOSSA - GAL. FERRI 7
• 32 BIT - VIA C. BATTISTI 14
• ELET. di BASSO - V.LE RISORGIMENTO 69

Provincia di Mantova
• CLICK - ON COMPUTER - S.S. GOITESE 168 - GOITO

Pavia
• POLIWARE - C.SO C. ALBERTO 76
• SENNA GIANFRANCO - VIA CALCHI 5

Provincia di Pavia
• A. FERRARI - C.SO CAVOUR 57 - MORTARA
• LOGICA MAINT - V.LE M.TE GRAPPA 32 - VIGEVANO
• M. VISENTIN - C.SO V. EMANUELE 76 - VIGEVANO

Sondrio
• CIPOLLA MAURO - VIA TREMOGGE 25

Provincia di Sondrio
• FOTONOVA - VIA VALERIANA 1 - S.PIETRO DI BERBENNO

Varese
• ELLE - EFFE - VIA GOLDONI 35
• IL C.TRO ELET. - VIA MORAZZONE 2
• SUPERGAMES - VIA CARROBBIO 13

Provincia di Varese
• BUSTO BIT - VIA GAVINANA 17-BUSTO A.
• MASTER PIX - VIA S.MICHELE 3 - BUSTO A.
• PUNTO UFFICIO - VIA R.SANZIO 8 - GALLARATE
• GRANDI MAGAZZINI BOSSI - VIA CLERICI 196 - GERENZANO
• J.A.C. - C.so MATTEOTTI 38 - SESTO C.

PIEMONTE

Alessandria
• BIT MICRO - VIA MAZZINI 102
• SERV. INFOR. - VIA ALESSANDRO III 47

Provincia di Alessandria
• SONY ITALIANA - VIA G. MANARA 7 - CASALE MONFERRATO
• SGE ELETTRONICA - VIA BANDELLO 19 - TORTONA
• COMPUTER TEMPLE - VIA F. CAVALLOTTI 13 - VALENZA

Asti
• ASTI GAMES - C.SO ALFIERI 26
• RECORD - C.SO ALFIERI 166/3 (Galleria Argenta)

Cuneo
• ROSSI COMPUTERS - C.SO NIZZA 42

Provincia di Cuneo
• PUNTO BIT - C.SO LANGHE 26/C - ALBA
• BOSETTI - VIA ROMA 149 - FOSSANO
• COMPUTERLAND - VIA MAZZINI 30/32 - SALUZZO

Novara
• PROGRAMMA 3 - V.LE BUONARROTI 8
• PUNTO VIDEO - C.so RISORGIMENTO 39/B

Provincia di Novara
• COMPUTER - VIA MONTE ZEDA 4 -ARONA
• ALL COMPUTER - C.SO GARIBALDI 106 - BORGOMANERO
• S.P.A. - C.SO DISSEGNA 21/BIS - DOMODOSSOLA
• ELLIOTT COMPUTER SHOP - VIA DON MINZONI 32 - INTRA
• TRISCONI VALERIA - VIA MAZZINI 90 - OMEGNA

Torino
• ABA ELETTRONICA - VIA C. FOSSATI 5/P
• ALEX COMPUTER E GIOCHI - C.SO FRANCIA 333/4
• COMPUTER HOME - VIA SAN DONATO 46/D
• COMPUTING NEW - VIA M. POLO 40/E
• C.D.M. ELETTR. - VIA MAROCHETTI 17
• DE BUG - C.SO V. EMANUELE II 22
• DESME UNIVERSAL - VIA S.SECONDO 95
• FDS ALTERIO - VIA BORGARO 86/D
• IL COMPUTER - VIA N. FABRIZI 126
• MICRONTEL - C.SO D. degli ABRUZZI 28
• PLAY GAMES SHOP - VIA C. ALBERTO 39/E
• RADIO TV MIRAFIORI - C.SO UNIONE SOVIETICA 381
• SMT ELETTRONICA - VIA BIBIANA 83/bis

Provincia di Torino
• PAUL E CHICO VIDEOSOUND - VIA V.EMANUELE 52 - CHIERI
• BIT INFORMATICA - VIA V. EMANUELE 154 - CIRIE'
• HI - FI CLUB - C.SO FRANCIA 92C - COLLEGNO
• MISTER PERSONAL - VIA CATTANEO 52 - FAVRIA
• I.C.S. - VIA TORINO 73 - IVREA
• DAG - VIA I MAGGIO 40 - LUSERNA S. GIOVANNI
• EUREX - C.SO INDIPENDENZA 5 - RIVAROLO CANAVESE
• DIAM INFORMATICA - C.SO FRANCIA 146 bis - RIVOLI
• FULLINFORMATICA - VIA V.VENETO 25 - RIVOLI
• GAMMA COMPUTER - VIA CAVOUR 3A-3B - SET.TORINESE

Vercelli
•ELETTROGAMMA - C.SO BORMIDA 27 ang. V.Montanara
• ELETTRONICA - STRADA TORINO 15

Provincia di Vercelli
• C.S.I. TEOREMA - VIA LOSANA 9 - BIELLA
• SIGEST - VIA BERTODANO 8 - BIELLA
• REMONDINO FRANCO - VIA ROMA 5 - BORGOSESIA
• FOTOSTUDIO TREVISAN - VIA XXV APRILE 24/B - COSSATO
• STUDIO FOTOGRAFICO IMARISIO - P.ZZA M. LIBERTA' 7 - TRINO

VENETO

Belluno
• UP TO DATE - VIA V. VENETO 43

Provincia di Belluno
• GUERRA COMPUTERS - V.LE MAZZINI 10/A - FELTRE

Padova
• BIT SHOP - VIA CAIROLI 11
• COMPUMANIA - VIA T. CAMPOSANPIERO 37
• D.P.R. DE PRATO R. - V.LO LOMBARDO 4
• G.F. MARCATO - VIA MADONNA DELLA SALUTE 51/53
• SARTO COMPUTER - VIA ARMISTIZIO 79

Provincia di Padova
• COMPUTER SERVICE - BORGO TREVISO 150 - CITTADELLA

Treviso
• BIT 2000 - VIA BRANDOLINI D'ADDA 14
• GUERRA EGIDIO & C. - V.LE CAIROLI 95

Provincia di Treviso
• DE MARIN COMPUTERS - VIA MATTEOTTI 142 - CONEGLIANO
• SIDESTREET - VIA SALVO D'ACQUISTO 8 - MONTEBELLUNA
• FALCON ELETTROAUDIOVIDEO - VIA TERRAGGIO 116 - PREGANZIOL

Venezia
• GUERRA EGIDIO & C. - VIA BISSUOLA 20/A - MESTRE
• TELERADIO FUGA - SAN MARCO 3457

Provincia di Venezia
• GUERRA EGIDIO & C. - VIA VIZZOTTO 29 - SAN DONA' DI PIAVE
• REBEL - VIA F. CRISPI 10 - SAN DONA' DI PIAVE

Verona
• CASA DELLA RADIO - VIA CAIROLI 10
• TELESAT - VIA VASCO DE GAMA 8

• Provincia di Verona
• UBER - CP 0363(RAG.SOC. DERTA) - VIA MASCAGNI 31 -CASTEL D'AZZANO
• FERRARIN - VIA DEI MASSARI 10 - LEGNAGO
• COMPUTERS CENTER - VIA CANTORE 26 - VILLAFRANCA

Vicenza
• ELET. BISELLO - V.LE TRIESTE 427/429
• SCALCHI MARKET - VIA CA' BALBI 139

Provincia di Vicenza
• SCHIAVOTTO - VIA ZANELLA 21 - CAVAZZALE
• GUERRA E. & C. - V.LE DELLE INDUSTRIE - MONTECCHIO MAGGIORE

FRIULI VENEZIA GIULIA

Gorizia
• E.CO. ELETTRONICA - VIA F.LLI COSSAR 23

Trieste
• AVANZO GIACOMO - P.ZZA CAVANA 7
• COMPUTER SHOP - VIA P. RETI 6
• COMPUTIGI - VIA XX SETTEMBRE 51
• CTI - VIA PASCOLI 4

Udine
• MOFERT 2 - VIA LEOPARDI 21
• R.T. SISTEM UDINE - VIA L. DA VINCI 99

Provincia di Udine
• IL PUNTO ELETTRONICO - VIA VENDRAMIN 184 - LATISANA
• IDRENO MATTIUSSI &C. - VIA LICINIANA 58 - MARTIGNACCO

TRENTINO ALTO ADIGE

Bolzano
• COMPUTER POINT - VIA ROMA 82/A
• MATTEUCCI PRESTIGE - VIA MUSEO 54

Provincia di Bolzano
• RADIO MAIR-ELECTRO - VIA CENTRALE 70 - BRUNICO
• ELECTRO RADIO HENDRICH - VIA DELLE CORSE 106 - MERANO
• ERICH KONTSCHIEDER - PORTICI 313 - MERANO
• ELECTRO TAPPEINER - P.ZZA PRINCIPALE 90 - SILANDRO

Trent0
• CRONST - VIA G. GALILEI 25

Provincia di Trento

• AL RISPARMIO - C.SO VERONA 138 - ROVERETO

LIGURIA

Genova

• ABM COMPUTER - P.ZZA DE FERRARI 24 rosso
• CAPRIOTTI G. - IA MAMIANI 4r - SAMPIERDARENA
• C.tro ELET. - VIA CHIARAVAGNA 10 R - VIA SESTRI 69R
• COM.le SOTTORIPA - VIA SOTTORIPA 115/117
• FOTOMONDIAL - VIA DEL CAMPO 3-5-9-11-13 r
• LA NASCENTE - VIA SAN LUCA 4/1
• PLAY TIME - VIA GRAMSCI 3/5/7 rosso
• RAPPR-EL - VIA BORGORATTI 23 R

Imperia

• CASTELLINO - VIA BELGRANO 44

Provincia di Imperia

• CENTRO HI-FI VIDEO - VIA DELLA REPUBBLICA 38 -SANREMO
• CASTELLINO - VIA GENOVA 48 - VENTIMIGLIA

La Spezia

• I.L. ELETTRONICA - VIA V. VENETO 123

Provincia di La Spezia

• I.L. ELETTRONICA - VIA AURELIA 299 - FORNOLA DI VEZZANO

Savona

• CASTELLINO - C.SO TARDY E BENECH 101

Provincia di Savona

• CELESIA ENZA - VIA GARIBALDI 144 - LOANO

EMILIA

Bologna

• EUROELETTRICA - VIA RANZANI 13/2
• MINNELLA ALTA FEDELTA' - VIA MAZZINI 146/2
• MORINI & FEDERICI - VIA MARCONI 28/C
• STERLINO - VIA MURRI 73/75

Provincia di Bologna

• S.C. COMPUTERS - VIA E. FERMI 4 - CASTEL SAN PIETRO
• S.P.E. INFORMATICA - VIA DI MEZZO PONENTE 385 - CREVALCORE
• ARCHIMEDE SISTEMI - VIA EMILIA 124 - S. LAZZARO DI SAVENA

Modena

• CO - EL - VIA CESARI 7
• ORSA MAGGIORE - P.ZZA MATTEOTTI 20
• VIDEO VAL WILLY COMPUTERS - VIA CANALETTO 223

Provincia di Modena

• NEW MEDIA SYSTEM - VIA ROMA 281 - SOLIERA

Parma

• BABARELLI G. - VIA B. PARENTE 14/A/B

Provincia di Parma

• PONGOLINI - VIA CAVOUR 32 - FIDENZA

Piacenza

• COMPUTER LINE - VIA G. CARDUCCI 4
• DELTA COMPUTER - VIA M. DELLA RESISTENZA 15/G

TEGGIO EMILIA

• COMPUTERLINE - VIA SAN ROCCO 10/C
• POOL SHOP - VIA EMILIA S. STEFANO 9/C

Provincia di Reggio Emilia

• MACCHIONI - VIA STATALE 467 - CASALGRANDE

ROMAGNA

Ferrara

• BUSINESS POINT - VIA CARLO MAYER 85

Forlì

• COMPUTER VIDEO CENTER - VIA CAMPO DI MARTE 122

Provincia di Forlì

• TOP BIT - VIA VENETO 12 - FORLIMPOPOLI
• COMPUTER HOUSE - V.LE TRIPOLI 193/D - RIMINI
• EASY COMPUTER - VIA LAGOMAGGIO 50 - RIMINI

REPUBBLICA S. MARINO

Ravenna

• COMPUTER HOUSE - VIA TRIESTE 134

Provincia di Ravenna

• ARGNANI - P.ZZA DELLA LIBERTA' 5/A - FAENZA
• ELECTRON INFORMATICA - VIA F.LLI CORTESI 17 - LUGO
• P.L.Z. INFORMATICA - P.ZZA SERCOGNANI 6 - FAENZA

TOSCANA

Arezzo

• DELTA SYSTEM - VIA PIAVE 13

Firenze

• ATEMA - VIA BENEDETTO MARCELLO 1a-1b
• ELETTRONICA CENTOSTELLE - VIA CENTO STELLE 5/a-b
• HELP COMPUTER - VIA DEGLI ARTISTI 15-A
• TELEINFORMATICA TOSCANA -VIA BRONZINO 36

Provincia di Firenze

• WAR GAMES - VIA R. SANZIO 126/A - EMPOLI
• NEW EVM COMPUTER - VIA DEGLI INNOCENTI 2 - FIGLINE VALDARNO
• C.tro INFOR. - VIA ZNOJMO 41 - PONTASSIEVE
• COSCI F.LLI - VIA ROMA 26 - PRATO
• BARBAGLI C. ELET. - VIA F. BONI 80 - PRATO

Grosseto

• COMPUTER SERVICE - VIA DELL'UNIONE 7

Livorno

• ETA BETA - VIA SAN FRANCESCO 30
• FUTURA 2 - VIA CAMBINI 19

Provincia di Livorno

• PUNTO ROSSO - VIA BARONTINI 28 - PIOMBINO

Provincia di Lucca

• IL COMPUTER - V.LE COLOMBO 216 - LIDO DI CAMAIORE
• SANTI VITTORIO - VIA ROMA 23 - S. ROMANO GARFAGNANA
• TOP GAMES - VIA S. ANDREA 122 - VIAREGGIO

Massa

• EURO COMPUTER - P.ZZA G. BERTAGNINI 4

Carrara

• RADIO LUCONI - VIA ROMA 24/B

Pisa

• ELECTRONIC SERVICE - VIA DELLA VECCHIA TRANVIA 10
• PUCCINI S.- CP 1199 (RAG.SOC. MAREX) - VIA C.CAMMEO 64
• TONY HI-FI - VIA CARDUCCI

Provincia di Pisa

• M.C. INFORMATICA - VIA DEL CHIESINO 4 - PONTEDERA (PI)

Pistoia

• ELECTRONIC SHOP - VIA DEGLI SCALZI 3

Provincia di Pistoia

• ZANNI &C. - C.SO ROMA 45 - MONTECATINI T.

Siena

• R. BROGI - P.ZZA GRAMSCI 28
• VIDEO MOVIE - VIA GARIBALDI 17

Provincia di Siena

• ELETTRONICA di BIFOLCHI - VIA DI GRACCIANO NEL CORSO 111 - MONTEPULCIANO

LAZIO

• CENTRO INF. - D.R.R. srl - TEL. 06-5565672

UMBRIA

Perugia

• MIGLIORATI - VIA S. ERCOLANO 3-10

Provincia di Perugia

• COMPUTER STUDIO'S - VIA IV NOVEMBRE 18/A - BASTIA UMBRA
• WARE - VIA DEI CASCERI 31 - CITTA'DI CASTELLO

Terni

• CGS SOFTWARE HOUSE - VIA DONIZETTI 71/A

BASILICATA

Matera

• G. GAUDIANO ELECTRONICS - VIA ROMA ang. XX SETTEMBRE 1

PUGLIA

Bari

• ARTEL - VIA GUIDO D'ORSO 9
• COMPUTER'S ARTS - V.LE MEUCCI 12/B
• PAULICELLI S. & F. - VIA FANELLI 231/C

Provincia di Bari

• F. FAGGELLA - C.SO GARIBALDI 15 - BARLETTA
• G.FAGGELLA - P.ZZA D'ARAGONA 62A - BARLETTA
• LONUZZO G. - VIA NIZZA 21 - CASTELLANA
• TECNOUFF. - VIA RICASOLI 54 - MONOPOLI
• TANGORRA N. - C.SO V.EMANUELE 130/B - TRIGGIANO

Brindisi

• MARANGI E NICCOLI - VIA PROV. SAN VITO 165

Provincia di Brindisi

• MILONE G. - VIA S.F. D'ASSISI 219 - FRANCAVILLA FONTANA

Foggia

• BOTTICELLI G. - VIA SAV POLLICE 2
• E.C.I. COMPUTER - VIA ISONZO 28
• LA TORRE - V.LE MICHELANGELO 185

Provincia di Foggia

• IL DISCOBOLO - VIA T. SOLIS 15 - SAN SEVERO

Lecce

• BIT - VIA 95 REGG.NTO FANTERIA 87/89

Provincia di Lecce

• TECNO UFFICIO - P.ZZA GIOVANNI XXIII 10 - GALLIPOLI
• CEDOK INFORMATICA - VIA UMBERTO I 116 - TRICASE

Taranto

• ELETTROJOLLY C.tro - VIA DE CESARE 13
• TEA - TEC. ELET. AV. - VIA R. ELENA 101

CAMPANIA

Provincia di Avellino

• FLIP FLOP - VIA APPIA 68 - ATRIPALDA

Benevento

• E.CO. INF. - VIA PEPICELLI 21/25

Caserta

• ENTRY POINT - VIA COLOMBO 31
• O.P.C. - VIA G. M. BOSCO 24

Provincia di Caserta

• M.P. COMPUTER - VIA NAPOLI 30 - MADDALONI
• DAMIANO - C.SO V. EMANUELE 23 -ORTA DI ATELLA
• FUSCO B. - VIA NAPOLI 24 - VAIRANO PATERNORA (FRAZ. VAIRANO SCALO)
• LINEA CONTABILE - VIA OSPEDALE 72/76 - SESSA A. (CE)

Napoli

• BABY TOYS - VIA CISTERNA DELL'OLIO 5/BIS
• CASA MUSICALE RUGGIERO - P.ZZA GARIBALDI 74 (INT. STAZ. F.F. S.S.)
• C.tro ELET. CAMPANO - VIA EPOMEO 121
• CI.AN - GALLERIA VANVITELLI 32
• CINE NAPOLI - VIA S. LUCIA 93/95
• DARVIN - CALATA SAN MARCO 26
• GIANCAR 2 - P.ZZA GARIBALDI 37
• ODORINO - L.GO LALA 22 A-B
• R 2 - VIA F. CILEA 285
• SAGMAR - VIA S. LUCIA 140
• TOP VIDEO - TOP COMPUTER - VIA S. ANNA DEI LOMBARDI 12
• VIDEOFOTOMARKET - VIA S. BRIGIDA 19

Provincia di Napoli

• ELECTRONIC DAY - VIA DELLE PUGLIE 17 - CASORIA
• TUFANO - S.S. SANNITICA 87 KM 7 - CASORIA
• SOF SUD - V.LE EUROPA 59 - CASTEL/MARE DI STABIA
• ELETTRONICA 2000 - C.SO DURANTE 40 - FRATTAMAGGIORE
• SPADARO - VIA ROMANI 93 - MADONNA DELL'ARCO
• GATEWAY - VIA NAPOLI 68 - MUGNANO
• VISPINI & DI VUOLO - VIA A.ROSSI 4 - POMPEI
• SPY CASH & CARRY - P.ZZA ARENELLA 6/A - NAPOLI
• NUOVA INFORMATICA SHOP - VIA LIBERTA' 185/191 - PORTICI
• BASIC COMPUTER - C.SO GARIBALDI 34 - POZZUOLI
• V.C. - C.SO SECONDIGLIANO 562/B - SECONDIGLIANO
• F. ELETTRONICA - VIA SARNO 102 - STRIANO
• TECNO - VIA V. VENETO 48 - TORRE DEL GRECO

Salerno

• COMPUMARKET - VIA BELVEDERE 35
• COMPUTER MARKET - C.SO VITTORIO EMANUELE 23

Provincia di Salerno

• KING COMPUTER - VIA OLEVANO 56 - BATTIPAGLIA
• DIMER POINT - V.LE AMENDOLA 36 - EBOLI
• IACUZIO F. - VIA MUNICIPIO 14 - MERCATO SAN SEVERINO
• COMPUTER SERVICE - VIA L.DA VINCI 81 - SCAFATI

CALABRIA

Catanzaro

• C.& G. COMPUTER - VIA F. ACRI 28
• PAONE S. & F. - VIA F. ACRI 93/99

Provincia di Catanzaro

• COMPUTER HOUSE - VIA BOLOGNA (L.GO OSPEDALE) - CROTONE
• RIOLO F.LLI - VIA VENEZIA 1/7 - CROTONE
• ING. FUSTO S. - C.SO NICOTERA 99 - LAMEZIA TERME

Cosenza

• MAISON DE L'INFORMATIQUE - VIA PASQUALE ROSSI 34/C
• SIRANGELO COMP. - VIA N. PARISIO 25

Provincia di Cosenza

• HI-FI ALFANO G. - VIA BALDACCHINI 109 - AMANTIA
• ELIGIO ANNICCHIARICO &C. - VIA ROMA 21 - CASTROVILLARI
• ALFA COMPUTER - VIA NAZIONALE 341/A - CORIGLIANO SCALO

REGGIO CALABRIA

• CONTROL SYSTEM - VIA S.F DA PAOLA 49 D
• SYSTEM HOU. - VIA FIUME ang. PALESTINO 1

Provincia di Reggio Calabria

• COMPUTER SHOP - V.LE MATTEOTTI 36/38 - LOCRI
• PICIEFFE - C.SO F. S. ALESSIO 19 - TAURIANOVA

SICILIA

• CENTRO INF. - ITALSOFT SRL - TEL. 0935-696090

SYSTEMS EDITORIALE PER TE

La voce III
Aggiunge al C/64 nuovi comandi Basic che consentono sia di far *parlare* il computer, sia di farlo *Cantare!* Diversi esempi allegati.
Cassetta: L. 12000 - Disco: L. 15000

Raffaello
Un programma completo per disegnare, a colori, con il C/64: linee, cerchi, quadrati, eccetera. Valido sia per disegno a mano libera che geometrico.
Cassetta: L. 10000

Oroscopo
Devi solo digitare la data di nascita e le coordinate geografiche del luogo che ti ha dato i natali. Vengono quindi elaborate le varie informazioni (case, influenze dei segni astrali, eccetera) e visualizzato un profilo del tuo carattere. Valido per qualsiasi anno, è indicato sia agli esperti sia ai meno introdotti. E' allegata una tabella delle coordinate delle più note città italiane e l'elenco delle ore legali in Italia dal 1916 al 1978.
Cassetta: L. 12000
Disco: L. 12000

Computer Music
Cassetta contenente numerosi brani di successo da far eseguire, in interrupt, al tuo C/64 sfruttando, fino in fondo, il suo generatore sonoro (SID).
Cassetta: L. 12000

Gestione Familiare
Il più noto ed economico programma per controllare le spese ed i guadagni di una famiglia.
Cassetta: L. 12000
Disco: L. 12000

Banca Dati
Il più noto ed economico programma per gestire dati di qualsiasi natura.
Cassetta: L. 12000
Disco: L. 12000

Matematica finanziaria
Un programma completo per la soluzione dei più frequenti problemi del settore.
Cassetta: L. 20000
Disco: L. 20000

Analisi di Bilancio
Uno strumento efficace per determinare con precisione i calcoli necessari ad un corretto bilancio.
Cassetta: L. 20000
Disco: L. 20000

Corso di Basic
Confezione contenente quattro cassette per imparare velocemente le caratteristiche delle istruzioni Basic del C/64 ed i rudimenti di programmazione. Interattivo.
Cassetta: L. 19000

Corso di Assembler
Un corso completo su cassetta per chi ha deciso di abbandonare il Basic del C/64 per addentrarsi nello studio delle potenzialità del microprocessore 6502. Interattivo.
Cassetta: L. 12000

Logo Systems
Il linguaggio più facile ed intuitivo esistente nel campo dell'informatica; ideale per far avvicinare i bambini al calcolatore. Diversi esempi allegati.
Cassetta: L. 6500

Compilatore Grafico Matematico
Uno straordinario programma compilatore, di uso semplicissimo, che permette di tracciare, sul C/64, grafici matematici Hi-Res ad altissima velocità. Esempi d'uso allegati.
Cassetta: L. 8000

Emulatore Ms-Dos e Gw-Basic
Un prodotto, unico nel suo genere, che permette di usare, sul C/64 dotato di drive, la sintassi tipica del più diffuso sistema operativo del mondo. Ideale per studenti.
Solo su disco: L. 25000

Emulatore Turbo Pascal 64
Permette di usare le più importanti forme sintattiche del linguaggio Turbo Pascal (anche grafiche!) usando un semplice C/64 dotato di drive. Ideale per studenti.
Disco: L. 25000

L.M. + Routine grafiche
Un fascicolo speciale (corredato di dischetto) suddiviso in due parti: corso completo di linguaggio macchina 6502 e implementazione di numerose routine che aggiungono al C/64 istruzioni Basic specifiche per la grafica, comprese quelle per disegnare in prospettiva!
Fascicolo + disco: L. 16000

Utility 1
Un dischetto pieno zeppo di programmi speciali per chi opera frequentemente con il drive.
Disco: L. 15000

Utility 2
Seconda raccolta di utility indispensabili per realizzare sofisticate procedure di programmazione.
Disco: L. 15000

Graphic Expander 128
Per usare il C/128 (in modo 128 e su 80 colonne) in modo grafico hi-res. Aggiunge nuove, potenti istruzioni Basic per disegnare in Hi-Res con la massima velocità in modalità 80 colonne.
Disco: L. 27000

Directory
Come è noto, a partire dal N. 10 di "Software Club" (la rivista su disco per l'utente dei "piccoli" computer Commodore), vengono riportati tutti i listati, in formato C/64-C/128, pubblicati su *"Commodore Computer Club"*. In precedenza tali listati venivano inseriti, mensilmente, in un dischetto, di nome "Directory", che oltre ai programmi di C.C.C. ospitava decine di altri file tra cui musiche nell'interrupt, giochi, listati inviati dai lettori e altro. Ogni disco, dal prezzo irrisorio, contiene quindi una vera miniera di software. Ordinando i dischetti di "Directory" si tenga conto che al N. 1 corrispondeva il contenuto del N. 34 di "Commodore Computer Club", al N. 2 il N. 35 e così via.
Ogni dischetto: L. 12.000

LIBRI TASCABILI

64 programmi per il C/64
Raccolta di programmi (giochi e utilità) semplici da digitare e da usare. Ideale per i principianti. ***(126 pag.)***
L. 4800

I miei amici C/16 e Plus/4
Il volumetto, di facile apprendimento, rappresenta un vero e proprio mini-corso di Basic per i due computer Commodore. Numerosi programmi, di immediata digitazione, completano la parte teorica. ***(127 pag.)***
L. 7000

62 programmi per C/16, Plus/4
Raccolta di numerosissimi programmi, molto brevi e semplici da digitare, per conoscere più a fondo il proprio elaboratore. Ideale per i principianti. ***(127 pag.)***
L. 6500

Micro Pascal 64
Descrizione accurata della sintassi usata dal linguaggio Pascal "classico". Completa il volume un programma di emulazione del PL/O sia in formato Microsoft sia in vesione C/64 (da chiedere, a parte, su disco). ***(125 pag.)***

L. 7000

Dal Registratore al Drive
Esame accurato delle istruzioni relative alle due più popolari periferiche del C/64. Diversi programmi applicativi ed esempi d'uso. ***(94 Pag.)***

L. 7000

Il linguaggio Pascal
Esame approfondito della sintassi usata nel famoso compilatore. ***(112 pag.)***

L. 5000

Utilities e giochi didattici
Raccolta di numerosi programmi, in versione C/64 e Spectrum, di particolare interesse per chi intenda svliluppare software didattico. ***(127 pag.)***

L. 6500

Simulazioni e test per la didattica
Raccolta di numerosi programmi che approfondiscono e tendono a completare la trattazione già affrontata sul precedente volume. *(127 pag.)*

L. 7000

Dizionario del Personal Computer
Raccolta dei termini più diffusi nel campo professionale; dizionario inglese - italiano. (Edizione ridotta). ***(96 pag.)***

L. 8000

Dizionario dell'Informatica
Dizionario inglese italiano di tutti i termini usati nell'informatica. (Edizione completa). ***(385 pag.)***

L. 20000

Word processing: istruzioni per l'uso
Raccolta delle principali istruzioni dei più diffusi programmi di w/p per i sistemi Ms-Dos: Word Star, Samna, Multimate Advantage, Word 3. ***(79 pag.)***

L. 5000

Telefax
Volumetto divulgativo sull'importanza del Telefax e sulle sue modalità operative caratteristiche. ***(66 pag.)***

L. 5000

Come compilare un giornale aziendale in Azienda
I principali problemi per chi opera in ambiente DPT, affrontati e risolti con la massima chiarezza e semplicità. ***(80 pag.)***

L. 5000

Unix
Un volumetto per saperne di più sul sistema operativo professionale per eccellenza. ***(91 pag.)***

L. 5000

ABBONAMENTO

Commodore Computer Club

11 fascicoli: L. 50.000

ARRETRATI

Ciascun numero arretrato di C.C.C. L. 5000

COME RICHIEDERE I PRODOTTI SYSTEMS

Coloro che desiderano procurarsi i prodotti della Systems Editoriale devono inviare, oltre alla cifra risultante dalla somma dei singoli prodotti, L. 3500 per spese di imballo e spedizione, oppure L. 6000 se si desidera la spedizione per mezzo raccomandata.
Le spese di imballo e spedizione sono a carico della Systems se ciascun ordine è pari ad almeno L 50000.

Per gli ordini, compilare un normale modulo di C/C postale indirizzato a:

C/C Postale N. 37 95 22 07
Systems Editoriale
Viale Famagosta, 75
20142 MILANO

Non dimenticate di indicare chiaramente, sul retro del modulo (nello spazio indicato con "Causale del versamento") non solo il vostro nominativo completo di recapito telefonico, ma anche i prodotti desiderati ed il tipo di spedizione da effettuare.

Per sveltire la procedura di spedizione sarebbe opportuno inviare, a parte, una lettera riassuntiva dell'ordine effettuato, allegando una fotocopia della ricevuta del versamento.

Volendo una spedizione in contrassegno è necessario anticipare la cifra di L 10000 (diecimila), da inviare secondo le modalità prima indicate, indipendentemente dalla quantità di materiale richiesto, e da conteggiare, comunque, IN AGGIUNTA alla cifra risultante dall'ordine. (Si sconsiglia, pertanto, la richiesta di di prodotti in contrassegno)

Chi volesse ricevere più celermente la confezione deve inviare la somma richiesta mediante assegno circolare, oppure normale assegno bancario (non trasferibile o barrato due volte) intestato a:

Systems Editoriale
Milano

N. 11 - Lire 12.000

Commodore 64 Club

- Cover
- Mercenaries
- G.F.T.
- Out of deep
- Cambogia
- E/Basic

Gioca con noi per un abbonamento omaggio a Commodore Computer Club

Commodore Club
Dir. Resp.: A. Ro
Edizioni System
Via Mos (MI) - Tel. (02) 55600310
Reg. Trib. 2-84 MePe